(19)日本国特許庁（ＪＰ）　(12) **公開特許公報（Ａ）**　(11)特許出願公開番号

**特開平8－63442**

(43)公開日　平成８年(1996)３月８日

| (51)Int.Cl.[6] | 識別記号 | 庁内整理番号 | ＦＩ | 技術表示箇所 |
|---|---|---|---|---|
| Ｇ０６Ｆ　15/163 | | | | |
| | | | Ｇ０６Ｆ　15/16　３１０　Ｖ | |

審査請求　未請求　請求項の数４　ＯＬ　（全 49 頁）

(21)出願番号　特願平6－202071

(22)出願日　平成６年(1994)８月26日

(71)出願人　000004226
日本電信電話株式会社
東京都新宿区西新宿三丁目19番２号

(72)発明者　山田　茂樹
東京都千代田区内幸町１丁目１番６号　日本電信電話株式会社内

(72)発明者　丸山　勝己
東京都千代田区内幸町１丁目１番６号　日本電信電話株式会社内

(72)発明者　久保田　稔
東京都千代田区内幸町１丁目１番６号　日本電信電話株式会社内

(74)代理人　弁理士　磯村　雅俊　（外１名）

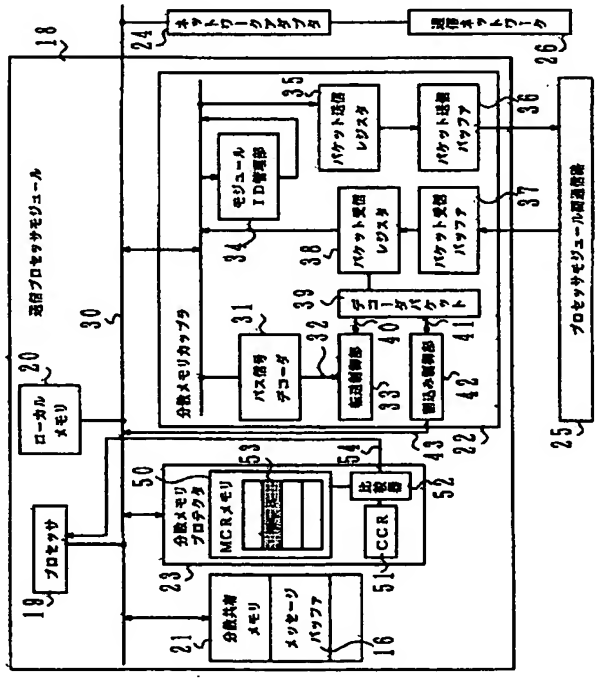
最終頁に続く

(54)【発明の名称】　マルチプロセッサシステム

(57)【要約】

【目的】　カーネル等のソフトウェアのオーバヘッドを削減し、かつ転送遅延時間が短く、処理効率の高いマルチプロセッサシステムのメッセージ転送を実現する。

【構成】　各プロセッサモジュールは分散共有メモリと分散メモリカップラとを備え、分散メモリカップラは、送信側の分散共有メモリに書き込みがあった時点で、アドレスを共有する他のプロセッサモジュールに書き込み情報を転送し、受信側のメモリにコピーする。ＭＢ管理マップを分散共有メモリに配置することにより、送信側分散共有メモリに書き込み、受信プロセッサモジュールが同一アドレスロケーションから読み出すだけで通信の同期をとることができる。返事は、ＭＢディスクリプタとＭＢ管理マップに書き込まれると、送信側が読み出すことにより、状態を送信側に伝達できる。また、受信側分散共有メモリをＦＩＦＯ構成にすることにより、処理要求の検出を簡単に行うことができる。



1

【特許請求の範囲】

【請求項１】複数のプロセッサモジュールから構成されたマルチプロセッサシステムにおいて、

上記プロセッサモジュールには、

各プロセッサモジュール間で共通のアドレスを有し、送信プロセッサモジュールと受信プロセッサモジュールの組み合わせで指定される管理単位毎に分割された分散共有メモリと、

上記送信プロセッサモジュールの分散共有メモリ（以下、送信側分散共有メモリ）とアドレスを共有する受信プロセッサモジュールの識別情報を記憶し、該送信側分散共有メモリへの書き込みが発生すると、上記受信プロセッサモジュールの識別情報で指定された受信プロセッサモジュールに、書き込みアドレスと書き込みデータを送信するとともに、受信した書き込みアドレスと書き込みデータをもとに受信プロセッサモジュールの分散共有メモリ（以下、受信側分散共有メモリ）の、送信側と同一のアドレスロケーションにコピーを行う分散共有メモリ制御手段と、

送信するメッセージの宛先情報をもとに受信プロセッサモジュールを特定し、上記分散共有メモリ上の送信プロセッサモジュールと受信プロセッサモジュールの組み合わせで指定されるメッセージバッファを捕捉するメッセージバッファ管理手段とを有し、

送信プロセッサモジュールから受信プロセッサモジュールにメッセージを送信する場合に、送信プロセッサモジュールは、上記メッセージバッファ管理手段により上記受信プロセッサモジュール対応の送信メッセージバッファを捕捉し、該送信メッセージバッファに上記メッセージを書き込み、

該送信プロセッサモジュールと受信プロセッサモジュールとが同一モジュールである場合には、上記受信プロセッサモジュールは、上記送信メッセージバッファから、直接、メッセージを読み出すことにより、同一プロセッサ内メッセージ通信を行い、

送信プロセッサモジュールと受信プロセッサモジュールが異なるモジュールである場合には、送信側の分散共有メモリ制御手段が、上記送信メッセージバッファのアドレスをもとに受信プロセッサモジュール識別情報を取り出し、該識別情報の受信プロセッサモジュールに送信メッセージバッファアドレスと書き込みデータを転送し、

受信側の分散共有メモリ制御手段は、受信した上記送信メッセージバッファのアドレスと上記書き込みデータをもとに受信側の分散共有メモリの、送信側と同一アドレスの受信メッセージバッファにコピーを行い、

上記受信プロセッサモジュールでは、上記受信メッセージバッファからメッセージを読み出すことにより、異なるプロセッサ間のメッセージ通信を行うことを特徴とするマルチプロセッサシステム。

【請求項２】請求項１に記載のマルチプロセッサシステムにおいて、上記受信側の分散共有メモリ制御手段は、さらにアドレスを共有する送信プロセッサモジュールの識別情報を記憶し、

上記送信プロセッサモジュールでは、該送信プロセッサモジュールと受信プロセッサモジュールの組み合わせで指定される分散共有メモリ上の送信制御エリアに、メッセージ制御情報あるいはメッセージバッファ管理情報を含む制御データを書き込み、

送信プロセッサモジュールと受信プロセッサモジュールとが同一モジュールである場合には、該受信プロセッサモジュールは、上記送信制御エリアから、直接、制御データを読み出してメッセージあるいはメッセージバッファに関する制御内容を認識し、対応する処理を実行した後、上記送信制御エリアに返答制御データを書き込み、

上記送信プロセッサモジュールは、上記送信制御エリアから上記返答制御データを読み出して受信側での処理結果を認識することにより、同一プロセッサモジュール内双方向通信を行い、

送信プロセッサモジュールと受信プロセッサモジュールが異なるモジュールである場合には、上記送信側の分散共有メモリ制御手段は、上記送信制御エリアのアドレスをもとに受信プロセッサモジュール識別情報を取り出し、該送信制御エリアアドレスと書き込みデータを上記受信プロセッサモジュールに転送し、

上記受信側の分散共有メモリ制御手段は、受信した上記送信制御エリアアドレスと上記書き込みデータをもとに受信側分散共有メモリの、送信側と同一アドレスの受信制御エリアにコピーを行い、

上記受信プロセッサモジュールは、上記受信制御エリアから制御データを読み出して、メッセージあるいはメッセージバッファに関する制御内容を認識し、対応する処理を実行した後、上記受信制御エリアに返答制御データを書き込むと、

受信側の分散共有メモリ制御手段は、上記受信制御エリアアドレスをもとに、上記送信プロセッサモジュール識別情報を取り出し、該識別情報をもとに上記受信制御エリアアドレスと上記返答制御データを上記送信プロセッサモジュールに転送し、

上記送信側の分散共有メモリ制御手段は、受信した上記受信制御エリアアドレスと上記返答制御データをもとに送信側分散共有メモリの、受信側と同一アドレスの送信制御エリアにコピーを行い、

上記送信プロセッサモジュールでは、上記送信制御エリアから上記返答制御データを読み出して、受信側での処理結果を認識することにより、プロセッサモジュール間双方向通信を行うことを特徴とするマルチプロセッサシステム。

【請求項３】複数のプロセッサモジュールから構成されたマルチプロセッサシステムにおいて、

上記プロセッサモジュールには、

3

各プロセッサモジュール間で共通の同一アドレスを有し、受信プロセッサモジュール対応の管理単位毎に分割された分散共有メモリと、
送信側分散共有メモリとアドレスを共有する受信プロセッサモジュールの識別情報を記憶し、送信側分散共有メモリへの書き込みが発生すると、上記受信プロセッサモジュール識別情報で指定された受信プロセッサモジュールに書き込みアドレスと書き込みデータを送信するとともに、受信した書き込みアドレスと書き込みデータをもとに受信側分散共有メモリの、送信側と同一のアドレスロケーションに書き込みを行う分散共有メモリ制御手段とを有し、
上記受信側の分散共有メモリのエリアはＦＩＦＯメモリで構成され、
複数の送信プロセッサモジュールから１つの受信プロセッサモジュールに処理要求を通知する際に、上記送信プロセッサモジュールと受信プロセッサモジュールが同一のモジュールである場合には、上記送信プロセッサモジュールでは、送信側分散共有メモリ上の自プロセッサモジュール宛処理要求ＦＩＦＯエリアに、処理要求データを書き込み、
上記送信プロセッサモジュールと受信プロセッサモジュールが異なるモジュールである場合には、上記送信プロセッサモジュールでは、上記送信側分散共有メモリ上の受信プロセッサモジュール宛処理要求ＦＩＦＯエリアに、処理要求データを書き込み、
送信側分散共有メモリ制御手段は、上記処理要求エリアのアドレスをもとに受信プロセッサモジュール識別情報を取り出し、上記処理要求エリアアドレスと書き込みデータを受信プロセッサモジュールに転送し、
受信側分散共有メモリ制御手段は、受信した上記処理要求ＦＩＦＯエリアアドレスと上記書き込みデータをもとに受信側分散共有メモリの、送信側と同一アドレスの上記処理要求ＦＩＦＯエリアに書き込むことにより、上記処理要求ＦＩＦＯエリアに複数の処理要求データを到着順に蓄積し、
上記受信側プロセッサモジュールでは、受信側分散共有メモリ上の上記要求ＦＩＦＯエリアから複数の上記処理要求データを順次読み出すことにより、処理要求を検出することを特徴とするマルチプロセッサシステム。
【請求項４】請求項１ないし３のいずれか１つに記載のマルチプロセッサシステムにおいて、上記送信側分散共有メモリ制御手段は、送信側分散共有メモリの連続する複数のアドレスロケーションに複数のデータが連続的に書き込まれた場合、上記複数のデータの先頭アドレスと複数のデータとをまとめて受信プロセッサモジュールに一括送信し、
受信側分散共有メモリ制御手段は、一括受信した上記先頭アドレスと上記複数のデータをもとに受信側分散共有メモリ上の同一アドレスロケーションに連続的にコピーを行うことを特徴とするマルチプロセッサシステム。

4

【発明の詳細な説明】
【０００１】
【産業上の利用分野】本発明は、複数のプロセッサモジュールから構成されるマルチプロセッサシステムに関し、特に転送遅延時間が短く、プロセッサモジュール間のリソース競合を回避でき、処理効率の高いメッセージ転送が可能なマルチプロセッサシステムに関するものである。
【０００２】
【従来の技術】複数のプロセッサを組み合わせてマルチプロセッサシステムを用いて、複数のオブジェクト（並列実行の単位となるプロセス）の間でメッセージを交信しながら処理を進めるオブジェクト指向分散処理方式が提案されている。例えば、T. Shimizu et al. “Low-Latency Communication Support for theAP1000”, Proceedings of the 19th International Symposium onComputer Architecture, pp. 288～297, 1992には、上記のようなオブジェクト指向分散処理方式が記述されている。図２および図３は、上記文献による従来例を示すシステム構成図およびプロセッサ間メッセージ通信方法を示すシーケンスチャートである。図２において、１－１，１－２はマルチプロセッサシステムにおけるプロセッサモジュール（ＰＭ）、２－１，２－２はプロセッサモジュール内のプロセッサ、３－１，３－２はそれぞれ対応するプロセッサ１－１，１－２から読み書きアクセスが可能なローカルメモリ、４－１，４－２はＰＭ間でメッセージ転送を行うためのＤＭＡ（Direct Memory Access）コントローラ、５はプロセッサ間通信路、６はローカルメモリ３－１上に設けられたメッセージバッファエリア、７は同じくローカルメモリ３－２上に設けられたメッセージバッファエリアである。
【０００３】図２に示すマルチプロセッサシステムにおいて、ＰＭ１－１からＰＭ１－２にメッセージを転送する場合のシーケンスを、図３により説明する。Ｓ１はメッセージバッファ要求、Ｓ２はメッセージバッファエリア６の確保、Ｓ３はメッセージバッファエリア６へメッセージを書込み、Ｓ４は送信要求、Ｓ５はＤＭＡコントローラ４－１に制御情報設定、Ｓ６はＤＭＡ起動、Ｓ７はメッセージバッファエリア６からプロセッサ間通信路５へ転送、Ｓ８はプロセッサ間転送、Ｓ９は受信一時バッファに記憶、Ｓ１０は割込み、Ｓ１１はメッセージバッファエリア７の確保、Ｓ１２はＤＭＡコントローラ４－２に制御情報設定、Ｓ１３はＤＭＡ起動、Ｓ１４は受信一時バッファからメッセージバッファエリア７に転送、Ｓ１５は割込み、Ｓ１６は受信先の特定、Ｓ１７は受信オブジェクト起動、Ｓ１８はメッセージの読出しの各ステップを示している。図３におけるプロセッサモジュール１－１の送信オブジェクト１０からメッセージバッファ要求Ｓ１があると、オペレーティングシステムの

中心的プログラムであるカーネル１１－１がステップＳ２でメッセージバッファエリア６をローカルメモリ３－１内に確保し、送信オブジェクト１０に通知する。送信オブジェクト１０は、ステップＳ３でメッセージバッファエリア６にメッセージを書き込む。図２においては、太線矢印Ｌ１０で示されている。送信オブジェクト１０はカーネル１１－１にステップＳ４で送信要求を出し、カーネル１１－１はステップＳ５でメッセージバッファエリア６の先頭アドレス（図２のＡＤＲ１）とメッセージサイズ（図２のｎ）、および転送先のプロセッサモジュール番号等の制御情報をＤＭＡコントローラ４－１に設定する。これは、図２においては、太線矢印Ｌ１１で示される。カーネル１１－１は、ステップＳ６でＤＭＡコントローラ４－１を起動し、ＤＭＡコントローラ４－１は設定された制御情報に従って、ステップＳ７，Ｓ８でメッセージをメッセージバッファエリア６から順次読み出し、プロセッサ間通信路５を経由して受信側ＰＭ１－２のＤＭＡコントローラ４－２に転送する。これは、図２では、太線矢印Ｌ１２で示される。

【０００４】ＤＭＡコントローラ４－２は、ステップＳ９でメッセージを一時記憶に蓄積すると同時に、並行してステップＳ１０でプロセッサ２－２のカーネル１１－２に割り込みをかける。これは、図２においては、太線矢印Ｌ１３で示される。カーネル１１－２は、ステップＳ１１で受信側ＰＭ１－２のローカルメモリ３－２上にメッセージバッファエリア７のアドレス（図２のＡＤＲ２）とメッセージサイズ（図２のｎ）等の制御情報を設定する。カーネル１１－２がステップＳ１３でＤＭＡ起動を行うと、ＤＭＡコントローラ４－２はステップＳ１４でメッセージをメッセージバッファエリア７に転送し（図２の太線矢印Ｌ１４）、ステップＳ１５でカーネル１１－２に転送終了を通知する。カーネル１１－２は、ステップＳ１６でメッセージ内に書込まれた宛先を見て、図２の受信オブジェクト１２を特定し、ステップＳ１７で起動する。受信オブジェクト１２がステップＳ１８でローカルメモリ３－２内のメッセージバッファ７からメッセージを読み出す（これは図２の太線矢印Ｌ１５）。以上の手順により、ＰＭ間のメッセージ転送が行われる。

【０００５】

【発明が解決しようとする課題】図３においては、送信オブジェクト１０がステップＳ３でメッセージバッファエリア６に書き込んだ段階で、受信ＰＭ宛のメッセージ転送をできるだけ早く開始することが望ましい。また、受信側においても、図３のステップＳ９でメッセージを受信した段階で、できるだけ早く受信オブジェクト１２にメッセージを引き渡すことが望ましい。しかしながら、従来の技術では、図３に示すように、ステップＳ５とステップＳ１２のＤＭＡ起動準備処理や、ステップＳ１１における受信側でのメッセージバッファエリアの捕



捉、ステップＳ１０，Ｓ１５における割り込み処理等があり、メッセージ送受信を終了させるまでには多くのカーネル処理が必要である。その結果、メッセージ転送遅延の増加とカーネルの処理オーバヘッド増加を招くという問題がある。特に、プロセッサ間通信路５の転送速度が大きい場合には、上記処理オーバヘッドによる転送遅延の比率は相対的に増加するため、処理オーバヘッドの削減と転送遅延時間の短縮が必須の条件となってくる。また、ステップＳ１０，Ｓ１５では、送信ＰＭからのメッセージ到着を通知する方法として割り込みを用いている。しかし、割り込み要求を受け付けるためには、現在実行中の情報を退避して、割り込み処理に切り替えるためのカーネルの処理オーバヘッドが一般に大である。従って、割り込み方式を多数のＰＭを含む超並列システムに適用した場合には、単位時間当りの割り込み回数の増加により、処理オーバヘッドが非常に大となって、転送遅延も増大するという問題が生じる。

【０００６】割り込みを回避する方法としては、従来より、ポーリング方式と呼ばれる要求受け付け方式が用いられている。ポーリング方式は、プロセッサ側が外部からの処理要求を予め決められたメモリやレジスタ等に記憶させておき、処理が可能になると、プロセッサが処理要求を読み出してきて処理を開始する方法である。このように、ポーリング方式は、割り込み方式に比較して処理切り替えのオーバヘッドが少ないが、処理要求を記憶する全てのメモリやレジスタを読み出す必要があるため、処理要求の発生頻度が少ない場合には処理要求の読み出しが空振りに終る、つまり処理要求が発生していない現象となることも多い。特に、多数のＰＭを有する超並列システムでは、処理要求の数および種類が多いため、それらの全てを定期的に読み出すための処理オーバヘッドが大きく、空振りの回数も増加して処理効率が低下するという問題があった。また、本出願人は本願に先立って、『分散共有メモリシステム』（特願平６－７４６６９号明細書および図面参照）を提案した。上記発明においては、分散共有メモリを送信プロセッサモジュールと受信プロセッサモジュールの組み合わせ毎に対応して分割し、各プロセッサモジュールからメッセージを送信する場合、自プロセッサモジュールを送信元とする受信プロセッサモジュール対応のエリアを使用する。その結果、異なる２つの送信プロセッサモジュールから同一受信プロセッサモジュールにメッセージ通信要求が同時に発生しても、互いに使用するメッセージバッファが重複することがなく、メッセージバッファ確保のための競合処理に伴う性能低下や性能ボトルネックを回避することができる。本発明の目的は、上記のような従来の課題を解決し、先願の発明を利用して、カーネル等のシステムソフトウェアのオーバヘッドを削減するとともに、転送遅延時間を短縮して、処理効率の高いメッセージ転送を実現できるマルチプロセッサシステムを提供すること

にある。

【０００７】

【課題を解決するための手段】上記目的を達成するため、本発明のマルチプロセッサシステムは、

（イ）複数のプロセッサモジュール（１８）から構成されたマルチプロセッサシステム（１７）において、上記プロセッサモジュール（１８）には、各プロセッサモジュール間で共通のアドレスを有し、送信プロセッサモジュールと受信プロセッサモジュールの組み合わせで指定される管理単位毎に分割された分散共有メモリ（２１）と、上記送信プロセッサモジュールの分散共有メモリ（以下、送信側分散共有メモリ２１）とアドレスを共有する受信プロセッサモジュールの識別情報を記憶し、該送信側分散共有メモリ（２１）への書き込みが発生すると、上記受信プロセッサモジュールの識別情報で指定された受信プロセッサモジュールに、書き込みアドレスと書き込みデータを送信するとともに、受信した書き込みアドレスと書き込みデータをもとに受信プロセッサモジュールの分散共有メモリ（以下、受信側分散共有メモリ２１）の、送信側と同一のアドレスロケーションにコピーを行う分散共有メモリ制御手段（２２）と、送信するメッセージの宛先情報をもとに受信プロセッサモジュールを特定し、上記分散共有メモリ上の送信プロセッサモジュールと受信プロセッサモジュールの組み合わせで指定されるメッセージバッファ（１６）を捕捉するメッセージバッファ管理手段（２００）とを有し、送信プロセッサモジュール（１８）から受信プロセッサモジュール（１８）にメッセージを送信する場合に、送信プロセッサモジュールは、上記メッセージバッファ管理手段（２００）により上記受信プロセッサモジュール対応の送信メッセージバッファ（１８０）を捕捉し、該送信メッセージバッファ（１８０）に上記メッセージを書き込み、該送信プロセッサモジュール（１８）と受信プロセッサモジュール（１８）とが同一モジュールである場合には、上記受信プロセッサモジュール（１８）は、上記送信メッセージバッファから、直接、メッセージを読み出すことにより、同一プロセッサ内メッセージ通信を行い、送信プロセッサモジュールと受信プロセッサモジュールが異なるモジュールである場合には、送信側の分散共有メモリ制御手段（２２）が、上記送信メッセージバッファのアドレスをもとに受信プロセッサモジュール識別情報を取り出し、該識別情報の受信プロセッサモジュールに送信メッセージバッファアドレスと書き込みデータを転送し、受信側の分散共有メモリ制御手段は、受信した上記送信メッセージバッファのアドレスと上記書き込みデータをもとに受信側の分散共有メモリの、送信側と同一アドレスの受信メッセージバッファにコピーを行い、上記受信プロセッサモジュールでは、上記受信メッセージバッファからメッセージを読み出すことにより、異なるプロセッサ間のメッセージ通信を行うことを特徴



としている。

【０００８】（ロ）上記受信側の分散共有メモリ制御手段は、さらにアドレスを共有する送信プロセッサモジュールの識別情報を記憶し、上記送信プロセッサモジュールでは、該送信プロセッサモジュールと受信プロセッサモジュールの組み合わせで指定される分散共有メモリ上の送信制御エリアに、メッセージ制御情報あるいはメッセージバッファ管理情報を含む制御データを書き込み、送信プロセッサモジュールと受信プロセッサモジュールとが同一モジュールである場合には、該受信プロセッサモジュールは、上記送信制御エリアから、直接、制御データを読み出してメッセージあるいはメッセージバッファに関する制御内容を認識し、対応する処理を実行した後、上記送信制御エリアに返答制御データを書き込み、上記送信プロセッサモジュールは、上記送信制御エリアから上記返答制御データを読み出して受信側での処理結果を認識することにより、同一プロセッサモジュール内双方向通信を行い、送信プロセッサモジュールと受信プロセッサモジュールが異なるモジュールである場合には、上記送信側の分散共有メモリ制御手段は、上記送信制御エリアのアドレスをもとに受信プロセッサモジュール識別情報を取り出し、該送信制御エリアアドレスと書き込みデータを上記受信プロセッサモジュールに転送し、上記受信側の分散共有メモリ制御手段は、受信した上記送信制御エリアアドレスと上記書き込みデータをもとに受信側分散共有メモリの、送信側と同一アドレスの受信制御エリアにコピーを行い、上記受信プロセッサモジュールは、上記受信制御エリアから制御データを読み出して、メッセージあるいはメッセージバッファに関する制御内容を認識し、対応する処理を実行した後、上記受信制御エリアに返答制御データを書き込むと、受信側の分散共有メモリ制御手段は、上記受信制御エリアアドレスをもとに、上記送信プロセッサモジュール識別情報を取り出し、該識別情報をもとに上記受信制御エリアアドレスと上記返答制御データを上記送信プロセッサモジュールに転送し、上記送信側の分散共有メモリ制御手段は、受信した上記受信制御エリアアドレスと上記返答制御データをもとに送信側分散共有メモリの、受信側と同一アドレスの送信制御エリアにコピーを行い、上記送信プロセッサモジュールでは、上記送信制御エリアから上記返答制御データを読み出して、受信側での処理結果を認識することにより、プロセッサモジュール間双方向通信を行うことも特徴としている。

【０００９】（ハ）複数のプロセッサモジュールから構成されたマルチプロセッサシステムにおいて、上記プロセッサモジュールには、各プロセッサモジュール間で共通の同一アドレスを有し、受信プロセッサモジュール対応の管理単位毎に分割された分散共有メモリと、送信側分散共有メモリとアドレスを共有する受信プロセッサモジュールの識別情報を記憶し、送信側分散共有メモリへ

の書き込みが発生すると、上記受信プロセッサモジュール識別情報で指定された受信プロセッサモジュールに書き込みアドレスと書き込みデータを送信するとともに、受信した書き込みアドレスと書き込みデータをもとに受信側分散共有メモリの、送信側と同一のアドレスロケーションに書き込みを行う分散共有メモリ制御手段とを有し、上記受信側の分散共有メモリのエリアはＦＩＦＯメモリで構成され、複数の送信プロセッサモジュールから１つの受信プロセッサモジュールに処理要求を通知する際に、上記送信プロセッサモジュールと受信プロセッサモジュールが同一のモジュールである場合には、上記送信プロセッサモジュールでは、送信側分散共有メモリ上の自プロセッサモジュール宛処理要求ＦＩＦＯエリアに、処理要求データを書き込み、上記送信プロセッサモジュールと受信プロセッサモジュールが異なるモジュールである場合には、上記送信プロセッサモジュールでは、上記送信側分散共有メモリ上の受信プロセッサモジュール宛処理要求ＦＩＦＯエリアに、処理要求データを書き込み、送信側分散共有メモリ制御手段は、上記処理要求エリアのアドレスをもとに受信プロセッサモジュール識別情報を取り出し、上記処理要求エリアアドレスと書き込みデータを受信プロセッサモジュールに転送し、受信側分散共有メモリ制御手段は、受信した上記処理要求ＦＩＦＯエリアアドレスと上記書き込みデータをもとに受信側分散共有メモリの、送信側と同一アドレスの上記処理要求ＦＩＦＯエリアに書き込むことにより、上記処理要求ＦＩＦＯエリアに複数の処理要求データを到着順に蓄積し、上記受信側プロセッサモジュールでは、受信側分散共有メモリ上の上記要求ＦＩＦＯエリアから複数の上記処理要求データを順次読み出すことにより、処理要求を検出することも特徴としている。

【００１０】（ニ）上記送信側分散共有メモリ制御手段は、送信側分散共有メモリの連続する複数のアドレスロケーションに複数のデータが連続的に書き込まれた場合、上記複数のデータの先頭アドレスと複数のデータとをまとめて受信プロセッサモジュールに一括送信し、受信側分散共有メモリ制御手段は、一括受信した上記先頭アドレスと上記複数のデータをもとに受信側分散共有メモリ上の同一アドレスロケーションに連続的にコピーを行うことも特徴としている。

【００１１】

【作用】本発明においては、①送信プロセッサモジュールの分散共有メモリ制御手段に、分散共有メモリアドレスを共有する受信プロセッサモジュール識別番号を記憶することにより、送信側共有メモリへの書き込みが行われると、分散共有メモリ制御手段からそのアドレスに対応する受信プロセッサモジュール識別番号を読み出し、対応する受信プロセッサモジュールに書き込み情報を送出すると、受信側の分散共有メモリ制御手段は、受信側分散共有メモリの同一アドレスロケーションにコピーを



行う。送信プロセッサモジュール内の分散共有メモリは、受信プロセッサ対応にメッセージバッファ群が分割されており、ある受信プロセッサ対応のメッセージバッファにメッセージが書き込まれると、分散共有メモリ制御手段により、対応する送受信プロセッサモジュール内の分散共有メモリの同一ロケーションのメッセージバッファにコピーが行われる。このように、プロセッサモジュール間で高速にオーバヘッドの少ないメッセージ通信が実現できる。

②また、受信側の分散共有メモリ制御手段にも、分散共有メモリアドレスを共有する送信プロセッサモジュール識別番号を記憶しておき、これにより受信側分散共有メモリのメッセージ制御エリアにメッセージ制御情報を書き込むと、送信側分散共有メモリにコピーされることにより、メッセージ制御に必要な双方向通信が行える。

【００１２】③さらに、受信側分散共有メモリの処理要求エリアをＦＩＦＯメモリにすることにより、送信側分散共有メモリに書き込んだ処理要求が、受信側分散共有メモリのＦＩＦＯに順次蓄積される。従って、複数の送信側プロセッサモジュールが同時に各分散共有メモリの同一ＦＩＦＯロケーションに処理要求を書き込んだ場合でも、受信側の分散共有メモリのＦＩＦＯで自動的に順序付けが行われるので、全ての処理要求がＦＩＦＯに順次書き込まれる。受信側は、分散共有メモリ内の要求登録ＦＩＦＯを順次読み出すだけで、発生した要求のみを効率よく取り出せる。従って、従来のポーリング処理や割り込み処理で必要とされた処理オーバヘッドを大幅に削減することができる。

④また、先願である特願平６－７４６６９号明細書および図面に記載のものに比較すると、分散共有メモリが全てのプロセッサモジュール間で共通のアドレスが付加され、送信プロセッサモジュールと受信プロセッサモジュールの組み合わせで指定される管理単位毎に分割されている点、および分散共有メモリを、自プロセッサモジュール内のローカル転送データのための送受信用と、異なるプロセッサモジュール間のリモート転送データのための送受信用のエリアに分割している点では共通している。しかし、上記明細書では、送信プロセッサモジュールが送信および受信メッセージバッファを捕捉し、受信プロセッサモジュールがそれらを解放し、受信プロセッサモジュールで再利用するか、あるいは送信プロセッサモジュールに再利用を許可する方法を用いている。その結果、２つのプロセッサモジュール間で双方向のメッセージ通信のトラヒックがアンバランスな場合、メッセージバッファの過不足が生じ易く、その制御手順も複雑で処理オーバヘッドも大きい。これに対して、本発明では、送信および受信メッセージバッファの捕捉、解放は送信プロセッサモジュールで一元的に行われ、送信側プロセッサモジュールで常に再利用されるため、制御手順も簡単である。また、上記明細書では、送信メッセージ

バッファから受信メッセージバッファへのコピーは、送信メッセージバッファにメッセージが書かれた後、メッセージ送信要求が発せられた時点で開始される。これに対して、本発明では、送信メッセージバッファから受信メッセージバッファへのコピーは、送信メッセージバッファにメッセージが書き込まれた瞬間から開始されるため、メッセージ転送遅延時間を短縮することができる。そして、本発明では、さらに受信側の分散共有メモリ制御手段にも、分散共有メモリアドレスを共有する送信プロセッサモジュール識別番号を記憶しておき、受信側分散共有メモリのメッセージ制御エリアにメッセージ制御情報を書き込むと、送信側分散共有メモリにコピーされる点が付加されている。また、分散共有メモリの処理要求エリアをＦＩＦＯメモリにする点も新たに追加されている。さらに、分散共有メモリ制御手段に登録されるプロセッサモジュール情報に応じて、一方向性通信、両方向性通信、放送型通信等の各種の通信パターンを任意に実現できる点でも追加されている。このように、本発明によれば、メッセージが送信側から受信側に伝達されるまでの遅延時間が大幅に削減されるとともに、一方向性、両方向性、放送型等の各種通信パターンを選択することができるので、柔軟性が高い。また、受信側プロセッサモジュールは受信側分散共有メモリの同じアドレスロケーションから制御データを読み出すだけで、プロセッサモジュール間の通信の同期を簡単にとることができ、かつ複数の送信プロセッサモジュールが同時に、分散共有メモリの同一番地に書き込みされた場合でも、受信側の分散共有メモリのＦＩＦＯに順次書き込まれるので、競合処理を全く必要としない。従って、受信側は、分散共有メモリ内の要求登録ＦＩＦＯを順次読み出すのみで、発生した要求のみを効率よく取り出すことができ、従来のようなサーチのための処理オーバヘッドを大幅に削減できる。

【実施例】以下、本発明の実施例を、図面により詳細に説明する。図４は、本発明の一実施例を示すマルチプロセッサシステムの構成図であって、３台のプロセッサからなる場合を示している。図４において、１７はマルチプロセッサシステム、１８－１，１８－２，１８－３はマルチプロセッサシステムにおけるプロセッサモジュール（ＰＭ）、１９－１，１９－２，１９－３は各ＰＭ内のプロセッサ、２０－１，２０－２，２０－３はそれぞれ対応するプロセッサ１９－１～１９－３から読み書きアクセスが可能なローカルメモリ、２１－１，２１－２，２１－３は全プロセッサモジュール間で共通のアドレスが付与された分散共有メモリである。ただし、プロセッサ１９－１からアクセスできる分散共有メモリはローカルメモリ２１－１のみであって、他のＰＭのローカルメモリ２１－２，２１－３にはアクセスできない。同じように、プロセッサ１９－２からアクセスできる分散共有メモリは２１－２のみであり、プロセッサ１９－３からアクセスできる分散共有メモリは２１－３のみである。２２－１，２２－２，２２－３は分散メモリカップラであって、これらは分散共有メモリ２１－１～２１－３にデータがそれぞれ書き込まれた時点で、その書き込みデータを、予め指定された他のプロセッサモジュールの全てないし１個に送信すると同時に、他のプロセッサモジュールから受信した書き込みデータを、分散共有メモリ２１－１～２１－３のうちの送信側と同一のアドレスロケーションに書き込むための装置である。



【００１３】次に、２３－１，２３－２，２３－３は分散メモリプロテクタであって、それぞれ、分散共有メモリ２１－１～２１－３内のメッセージバッファに記憶されているメッセージを不正なアクセスから保護する装置である。２４－，２４－２，２４－３はネットワークアダプタであって、通信ネットワーク２６を介して他の分散システムとメッセージ等を交換するための装置である。２５はプロセッサモジュール間通信路であって、マルチプロセッサシステム１７内のＰＭ間でメッセージあるいはそれらの制御情報を転送するための装置である。プロセッサモジュール間通信路の具体的実現手段としては、システムバス、ＬＡＮ、パケットスイッチングネットワーク、リングネットワーク等、送信側と受信側で１対１の通信ができるものであれば何でもよい。２６は複数のマルチプロセッサシステム１７間を相互接続するための通信ネットワークであって、ここでは転送データを固定長のブロックに分割して、それに行き先ヘッダを付加した５３バイトのセルにして、ネットワーク内を転送するＡＴＭ（Asynchronous Trasnsfer Mode)ネットワークを想定しているが、その他の種類の通信ネットワークでも適用可能であることは勿論である。

【００１４】図５、図６および図７は、図４における分散共有メモリのデータ配置例を示す図である。ここでは、データの配置を物理アドレスレベルで示している。分散共有メモリ内の通信領域は、①ＰＭ間通信エリア、②ＰＭ内通信エリア、③ＦＩＦＯ通信エリア、の３種類に大別される。このうち①ＰＭ間通信エリアは、マルチプロセッサシステム１７内のＰＭ間通信情報を記憶するエリアであり、②ＰＭ内通信エリアは、同一ＰＭ内で送受信される通信情報を記憶するエリアであり、また③ＦＩＦＯ（First In First Out)通信エリアは、ＰＭ間にまたがる処理要求をＦＩＦＯ形式で蓄積記憶するエリアである。ＰＭ間通信エリアおよびＰＭ内通信エリアは、

（ｉ）メッセージを記憶するメッセージバッファ（ＭＢ）エリア

（ii）メッセージバッファの捕捉／解放状態を表示するＭＢ管理マップエリア

（iii）メッセージバッファアドレス等の送受間引き継ぎ制御情報（ディスクリプタ）を複数個、連続アドレスに配置したディスクリプタリングエリア

13

それぞれに分けられる。これらのエリアの具体的使用方法については、追って説明する。上記ＭＢ、ＭＢ管理マップ、およびディスクリプタリングは、それぞれページを単位として物理メモリの割り当てが行われ、これらのページは仮想記憶のページテーブルにより管理されている。そして、ＭＢエリア用のページテーブルには、プロセッサのスーパバイザモード（カーネルやカーネルから起動される各種のシステム制御プログラムが実行されるモード）と、ユーザモード（各種アプリケーションプログラムが実行されるモード）との両方からアクセスが許されるように保護情報が設定されている。その結果、各種アプリケーションプログラムがＭＢエリアにメッセージを直接読み書きすることができるので、処理の効率化を図ることが可能となる。一方、ＭＢ管理マップ用ページと、ＭＢディスクリプタリング用ページのページテーブルには、プロセッサのスーパバイザモードのアクセスのみを許すように保護情報が設定されている。その結果、アプリケーションプログラムからＭＢ管理マップおよびＭＢディスクリプタリングへの不正書き込みに対して保護が行われる。

【００１５】図５～図７のエリア内の記述に示すように、分散共有メモリ２１上の各エリアを識別するために、Ｍｉｊ－ｋのような識別名を用いている。ここで、Ｍはプロセッサモジュール対応のＭＢ／ＭＢ管理マップ／ＭＢディスクリプタリング／ＦＩＦＯ通信エリアの識別記号であり、ＰＭ（１８－１）に対応するＭＢはＸ、そのＭＢ管理マップはＸＭ、ＭＢディスクリプタリングはＸＤと表現される。同じように、ＰＭ（１８－２）に対応するＭＢはＹ、そのＭＢ管理マップはＹＭ、ＭＢディスクリプタリングはＹＤ、ＦＩＦＯ通信エリアはＹＦ、ＰＭ（１８－３）に対応するＭＢはＺ、そのＭＢ管理マップはＺＭ、ＭＢディスクリプタリングはＺＤ、ＦＩＦＯ通信エリアはＺＦのように名称が付けられている。また、ｉ，ｊは、それぞれ送信ＰＭの識別番号（ＰＭ　ＩＤ）と受信ＰＭの識別番号（ＰＭＩＤ）を表わしており、ＰＭ＃１（１８－１）、ＰＭ＃２（１８－２）、ＰＭ＃３（１８－３）に対応してそれぞれ１，２，３の番号が付与されている。ただし、例外として、ＦＩＦＯ通信エリアＸＦ，ＹＦ，ＺＦは値ｊのみを有し、値ｉを持っていない。その理由としては、ＦＩＦＯ通信エリアが値ｊで表わされる受信側プロセッサモジュール対応に分割されており、値ｉで表わされる各送信プロセッサモジュール間で共通に使用するため、エリアの指定情報として値ｉを必要としないためである。

【００１６】ｋは同一種類のエリア内での複数個のＭＢ／ＭＢ管理マップ／ＭＢディスクリプタ／ＦＩＦＯ通信エリアを識別するためのもので、１から順に値が割り振られている。例えば、Ｚ１３－２は、分散共有メモリ２１－３上の、ＰＭ＃１（１８－１）からＰＭ＃３（１８－３）へのＭＢのうちの第２番目のＭＢを表わす。また、ＸＤ２１－２は、分散共有メモリ２１－１上の、ＰＭ＃２（１８－２）から１番目のＰＭ＃１（１８－１）へのＭＢディスクリプタのうち、第２番目のＭＢディスクリプタを表わす。図５～図７から明らかなように、ＰＭ間通信エリアではｉ≠ｊ、ＰＭ内通信エリアではｉ＝ｊとなる。ただし、例外として、ＭＢディスクリプタ／ＦＩＦＯ通信エリアで、送信処理要求情報が登録される場合にはＳと表現され、受信処理要求情報が登録される場合にはＲと表現される。例えば、ＸＤ１２－Ｒは、分散共有メモリ（２１－１）上のＰＭ＃１（１８－１）からＰＭ＃２（１８－２）への受信ディスクリプタを表わし、ＺＦ２－Ｓは、分散共有メモリ（２１－３）上のＰＭ＃２（１８－２）への受信処理ＦＩＦＯ通信エリアを表わしている。各分散共有メモリは、実効的に必要とされるメモリ量、つまり自プロセッサが送受信するために必要なエリアのみが実装されている。例えば、分散共有メモリ２１－１では、図５、図６のハッチ部分のみが実装されており、それ以外のエリアにはメモリが実装されていない。

14

【００１７】２つのメッセージバッファのｉ，ｊ，ｋの値がそれぞれ一致し、Ｍの値が異なるＭＢがペアを構成する。このペアは、異なる分散共有メモリの同一物理アドレスロケーションに配置される。例えば、図５～図７において、メッセージバッファＸ１２－１とＹ１２－１はペアを組んでおり、互いに同じ物理アドレスを有している。各ペアは、送信側ＰＭのカーネルにより動的に捕捉、解放が行われる。各ＭＢは固定長の大きさで構成されており、図５～図７に示すように、宛先ＰＭ－ＩＤが同一の複数のＭＢを、分散共有メモリの連続するアドレスロケーションに割り付ける。これを、宛先ＰＭ対応ＭＢプールと呼ぶ。各ＭＢプール対応にＭＢ管理マップが存在し、ＭＢプール内のＭＢ数に応じたＭＢ管理マップのエントリが用意され、それらが分散共有メモリの連続エリアに割り付けられている。例えば、図５において、メッセージバッファＸ１２－１，Ｘ１２－２は１つのＭＢプールを構成し、Ｘ１２－１に対してＭＢ管理マップエントリＸＭ１２－１が、Ｘ１２－２に対してＭＢ管理マップエントリＸＭ１２－２が対応する。『対応』という意味は、ＭＢの先頭アドレス（以下、単にＭＢアドレスと呼ぶ）が付与されると、そのＭＢ管理マップエントリアドレスを求めることができ、逆にＭＢ管理マップエントリアドレスからＭＢアドレスも求めることができることを意味している。ＭＢ管理マップの各エントリは、対応するＭＢが『未使用』か／『使用中』かの状態を表示し、カーネルにより値が設定される。ＭＢの場合と同じ考え方により、送信側分散共有メモリＭＢ管理マップエントリと、受信側分散共有メモリの同一アドレスのＭＢ管理マップエントリがペアを構成している。例えば、ＭＢ管理マップエントリＸＭ１２－２とＹＭ１２－２がペアを組んでいる。

15

【００１８】ディスクリプタリングエリアも、ＭＢと同じように、宛先ＰＭ－ＩＤが同一の複数のＭＢディスクリプタを分散共有メモリの連続するアドレスロケーションに割り付けており、割り付けた全体をディスクリプタリングと呼ぶ。『リング』と呼ばれている理由は、ディスクリプタリングに含まれる複数のディスクリプタを若いアドレス順にサイクリックに使用していくからである。送信側の各ディスクリプタリングと同一アドレスのエリアが、受信プロセッサモジュールの分散共有メモリにも配置される。ディスクリプタリングは、送信オブジェクトからのメッセージ制御情報を他のＰＭあるいは他のシステムに引き継ぐための『送信ディスクリプタリング』と、他のＰＭあるいは他のシステムから受信したメッセージ制御情報を受信オブジェクトに引き継ぐための『送信ディスクリプタリング』に分けられる。ディスクリプタリングもＭＢやＭＢ管理マップの場合と同じように、送信側と受信側の同一アドレス間でペアを構成する。ディスクリプタリング内のディスクリプタとＭＢとの対応は、実行時にカーネルにより動的に決定される。具体的には、例えばＰＭ＃１（１８－１）からＰＭ＃２（１８－２）にメッセージを送信する場合に、ＰＭ（１８－１）のカーネルが空きのＭＢプールの中から適当なＭＢ（例えばＸ１２－２）を捕捉し、次に対応する送信ディスクリプタリング（例えばＸＤ１２－Ｓ）を選択し、その中の『未使用』の最若アドレスのディスクリプタにＭＢアドレスを登録する。このような方法でディスクリプタとＭＢとの対応が動的に決定される。なお、送信ディスクリプタリングＸＤ１２－ＳとペアをＡ組む送信ディスクリプタリングはＹＤ１２－Ｓである。

【００１９】図７におけるＦＩＦＯ通信エリアは、送信オブジェクトからのメッセージ送信処理要求を他のＰＭまたは他のシステムに通知するための『送信処理ＦＩＦＯエリア』と、他のＰＭまたは他のシステムから受信したメッセージ受信処理要求を受信オブジェクトに通知するための『受信処理ＦＩＦＯエリア』に分けられる。このうち、メッセージ処理要求送信側の分散共有メモリは、通常のＲＡＭ（ランダムアクセスメモリ）で構成されるが、メッセージ処理要求受信側の分散共有メモリは、ＦＩＦＯで構成されている。例えば、ＰＭ＃１（１８－１）上のＰＭ＃１（１８－１）宛受信処理ＦＩＦＯエリアＸＦ１－Ｒは、他のＰＭからのメッセージ受信処理要求を記憶するためにＦＩＦＯメモリで構成されているが、ＰＭ＃２上のＰＭ＃１（１８－１）宛受信処理ＦＩＦＯエリアＹＦ１－Ｒおよび、ＰＭ＃３上のＰＭ＃１（１８－１）宛受信処理ＦＩＦＯエリアＺＦ１－Ｒは、いずれもＲＡＭで構成されている。この理由としては、受信処理要求を書き込む側のＰＭ（ＰＭ＃２とＰＭ＃３）が受信処理要求をそれぞれ受信処理ＦＩＦＯエリアＹＦ１－ＲとＺＦ１－Ｒに同時に書き込んだ場合、それらが同時に受信処理要求を読み出す側のＰＭ（ＰＭ＃

16

１）の受信処理ＦＩＦＯエリアＸＦ１－Ｒに到着するので、これらの受信処理要求を全て蓄積するためにＦＩＦＯメモリを使用するからである。

【００２０】図１は、本発明の一実施例を示すプロセッサモジュールの内部構成図である。図１のプロセッサモジュールは、図４におけるＰＭ１８－１～１８－３に該当している。なお、以下の説明では、３つのＰＭ内に存在する構成要素の各々を識別する必要がないときには、－１，－２，－３の各識別符号を省略する。例えば、個々の分散共有メモリを指定しないときには、分散共有メモリ２１－１，２１－２，２１－３と記述することなく、単に分散共有メモリ２１と記述することにする。図１において、１６は分散共有メモリ２１上に配置されたメッセージバッファ（ＭＢ）、３０はプロセッサモジュール（ＰＭ）１８内の各種装置を結合するためのプロセッサバス、２２は分散メモリカップラで、分散共有メモリ２１に実装されているエリアへの書き込みアクセスがあったとき、他のＰＭの分散共有メモリの同じアドレスロケーションにも書き替えデータを送信する機能、および他のＰＭから書き替えデータを受信して、自ＰＭの分散共有メモリへの書き込みを行う機能を有する。分散メモリカップラ２２は、以下の構成要素を含んでいる。先ずバス信号デコーダ３１であり、これはプロセッサバス３０の信号線上の信号を解読し、所属する分散共有メモリ２１への書き込みアクセスがあれば、信号線３２に‘１’を出力して、転送制御部３３を起動する。転送制御部３３は、分散メモリカップラ２２全体の制御を司る部分であり、ここから内部ロジックに各種制御信号（図示省略）を供給する。３４はモジュールＩＤ管理部であり、分散共有メモリ２１への書き込みデータをＰＭ間で送受する際に必要な宛先プロセッサモジュールＩＤ（ＰＭ　ＩＤ）を記憶している。モジュールＩＤ管理部３４の詳細については図６に示している。

【００２１】図１において、３５はパケット送信レジスタであって、他のＰＭに転送する分散共有メモリアドレスや書き込みデータ等、または他のＰＭへの割り込み情報等を記憶する。３６はパケット送信バッファであり、パケット送信レジスタ３５からのデータを受け取り、他のＰＭに転送するまでの一時待ち合わせ機能を有している。３７はパケット受信バッファであり、他のＰＭから転送されてきたデータを受け取り、パケット受信レジスタ３８に渡すまでの一時待ち合わせの機能を有する。パケット受信レジスタ３８は、他のＰＭから転送されてきた情報を記憶するレジスタであり、パケット送信レジスタ３５が送り出した内容と同じデータが記憶される。３９はパケットデコーダであり、パケット受信レジスタ３８内のデータをデコードして、分散共有メモリ２１内のアドレスロケーションへの書き込み要求であれば、信号線４０を介して転送制御部３３に分散共有メモリ書き込みの実行を依頼する。また、デコードの結果が、他のＰ

Mからの割り込みであれば、信号線４１を介して割込み制御部４２に割込み処理を依頼する。割込み制御部４２は、信号線４３を経由してプロセッサ１９に割り込み要求を行う。

【００２２】図１において、２３は分散メモリプロテクタであって、分散共有メモリ２１に配置されたＭＢ（１６）内のメッセージを不正なアクセスから保護するハードウェア機構である。分散メモリプロテクタ２３は、カレントケーパビリティレジスタ（ＣＣＲ：Curent Capability Register）５１、ＭＣＲメモリ（MemoryCapability Register Memory）５０、比較器５２を含んでいる。ＣＣＲ５１は、カレントケーパビリティを記憶するレジスタである。カレントケーパビリティは、具体的には、プロセッサ１９で実行中のアプリケーションオブジェクト（プロセッサのユーザモードで実行されるオブジェクト）のＩＤを意味している。ＭＣＲメモリ５０は、複数のＭＣＲ５３から構成され、各ＭＣＲ５３は分散共有メモリ２１上の各ＭＢ１６対応に、メモリケーパビリティを記憶する。メモリケーパビリティは、具体的には、対応するＭＢ１６へのメモリアクセスが生じると、ＭＢアドレスを基に対応するＭＣＲ５３をＭＣＲメモリ５０の中から選択し、これを取り出す。比較器５２は、選択されたＭＣＲ５３とＣＣＲ５１とを比較してメッセージバッファへの正しいアクセスであるか否かを判定し、不一致であれば不正アクセスであると判定する。不正アクセスを検出した場合には、並行して行われているＭＢ１６へのアクセスを中継し、信号線５４を介してプロセッサ１９に緊急通通する。

【００２３】以下、分散メモリプロテクタ２３を、ＰＭ１８－１内の送信オブジェクトからＰＭ１８－２内の受信オブジェクトへのメッセージ通信に適用した場合の制御手順について述べる。

（ａ）送信ＰＭ１８－１のカーネルは、送信オブジェクトの実行を開始する前に、カレントケーパビリティレジスタＣＣＲ５１－１にカレントケーパビリティつまり送信オブジェクトＩＤを設定する。

（ｂ）送信ＰＭ１８－１のカーネルは、分散共有メモリ２１－１上に送信ＭＢ１６－１を捕捉した後、送信ＭＢ１６－１対応のＭＣＲ５３－１にメモリケーパビリティを設定する。この時点のメモリケーパビリティは、ＭＢの捕捉を要求したオブジェクトのＩＤつまり送信オブジェクトＩＤである。

（ｃ）送信プロセッサ１９－１から送信ＭＢ１６－１へのメモリアクセスがある度毎に、ＭＣＲ５３－１の中からＭＢアドレスを基に対応するＭＣＲ５３－１を１個選択して、それをＣＣＲ５１－１と比較する。ＭＣＲ５３－１とＣＣＲ５１－１のオブジェクトＩＤの値が等しいときには正しいアクセスであり、等しくないときには不正アクセスであると判定する。

（ｄ）メッセージが受信ＰＭ１８－２の受信ＭＢ１６－２に到着したと仮定する。受信ＰＭ１８－２のカーネルは、受信オブジェクトの実行を開始する前に、ＣＣＲ５１－２にカレントケーパビリティつまり受信オブジェクトＩＤを設定する。



（ｅ）受信ＰＭ１８－２のカーネルは、受信ＭＢ１６－２に対応するＭＣＲ５３－２に、メモリケーパビリティを設定する。この時点のメモリケーパビリティは、メッセージの宛先であるオブジェクトのＩＤつまり受信オブジェクトＩＤである。

【００２４】（ｆ）受信プロセッサ１９－２から受信ＭＢ１６－２へのメモリアクセスがある度毎に、比較器５２－２によりＭＣＲ５３－２とカレントケーパビリティレジスタＣＣＲ５１－２内のオブジェクトＩＤの値を比較し、等しければ正しいアクセス、等しくなければ不正アクセスと判定する。

上記動作において、ＭＣＲ５３とＣＣＲ５１に、それぞれメモリケーパビリティとカレントケーパビリティが設定されている期間（プロテクションウィンドウ開放期間）は、ＭＢの所有権が付与された送信または受信オブジェクトのみがそのＭＢをアクセスできる。プロテクションウィンドウ開放期間中に、もし無関係のオブジェクトがＭＢアクセスを行うと、ＭＣＲとＣＣＲのケーパビリティの不一致が生じて、不正アクセスであることが検出される。プロテクションウィンドウ開放期間以外では、送信、受信オブジェクトも含めた全てのアプリケーションオブジェクトは、このＭＢにアクセスすることができないので、強固なメモリ保護が実現されることになる。なお、分散メモリプロテクタ２３は、プロセッサのアプリケーションモード（またはユーザモード）でのＭＢアクセスで動作するが、カーネルモード（またはスーパバイザモード）でのメモリアクセスは動作せず、無条件にメモリアクセスを許容する。この理由としては、一般にカーネルモードで実行されるプログラム（例えばカーネル）は、アプリケーションオブジェクトに比較して潜在バグの含有率が小さく、不正なメモリアクセスをする可能性が少ないからである。

【００２５】図８は、図１におけるモジュールＩＤ管理部の内部構成図である。図８において、６０はモジュールＩＤディレクトリであり、ＣＡＭ（Content Addressable Memory）セル部６１とデータメモリ部６２から構成される。ＣＡＭとは、検索キーデータを入力して、これと各ワード（ＣＡＭセル）の記憶データの内容を一斉に比較照合して、指定された検索条件に合致した内容のＣＡＭセルを選択表示するメモリである。ＣＡＭセルの各々には、『分散共有メモリページアドレス６３』が記憶されている。分散共有メモリページアドレス６３は、他のプロセッサモジュールと共用されているアドレスのページである。例えば、分散共有メモリ２１の連続エリアに配置された宛先ＰＭ対応のＭＢプールのページアドレス、そのＭＢ管理マップのページアドレス、ディスク

リプタリングのページアドレス、宛先ＰＭ対応のＦＩＦＯ通信エリアのページアドレス等が設定される。図５～図７で説明したように、ＭＢ、ＭＢ管理マップ、ディスクリプタリング、ＦＩＦＯ通信エリアは、それぞれ宛先ＰＭ毎に異なるページに割り付けられている。一方、データメモリ部６２は、各ＣＡＭセルのワードに対応する補助データを記憶するＲＡＭであり、分散共有メモリページアドレス６３を共有している宛先プロセッサモジュールのＩＤ（宛先ＰＭ　ＩＤ）６４を記憶している。

【００２６】ここで、ＭＢ管理マップ、ディスクリプタリングは、送信ＰＭ、受信ＰＭの両方から書き込みが行えるようにするため、送信ＰＭのモジュールＩＤディレクトリ６０には、〔共有ページアドレス、受信ＰＭ　ＩＤ〕のペアが記憶され、受信ＰＭＭのモジュールＩＤディレクトリ６０には、〔送信側と同じ共有ページアドレス、送信ＰＭ　ＩＤ〕のペアが記憶される。このように、各ＰＭのモジュールＩＤディレクトリ６０には、コピーをしたい全てのＰＭのＰＭ　ＩＤが登録されている。図８の６５はキーレジスタであり、これはＣＡＭセル部６１の内容と比較照合するための検索キーデータを記憶する。６６はマスクレジスタであり、これはキーレジスタ６５内の検索キーデータとＣＡＭセル部６１との比較を行う場合に、比較しないビット位置を指定するためのレジスタである。６７はＣＡＭコマンドレジスタであり、ＣＡＭセル部６１に関する各種指令を記憶する。６８は結果レジスタであり、これは検索した結果、附帯マＣＡＭセルの有無の表示、複数セル一致の有無の表示、空きセルの有無の表示等を行う部分である。キーレジスタ６５とＣＡＭセル部６１との比較処合の結果、一致したＣＡＭを使用しなくても、よく知られているようにハッシングの技術とＲＡＭとを組み合わせて、ＣＡＭと同じような機能を実現することができる。

【００２７】図８に示すモジュールＩＤ管理部３４は、以下のように動作する。分散共有メモリ２１への書き込みアクセスがあると、そのアドレスがキーレジスタ６５に格納され、そのページアドレスとＣＡＭセル部６１とが比較される。一致したセルが検出されると、そのアドレスが他のＰＭにより共用されていることを意味するので、データメモリ部６２から対応する宛先ＰＭ　ＩＤ６４を読み出し、これを分散共有メモリアドレスに付与した後、書き込みデータと合わせて送信パケットを作成し、図１に示すパケット送信レジスタ３５に設定する。パケット送信レジスタ３５からパケット送信バッファ３６を経由してパケットを宛先ＰＭに送出する。受信側の分散メモリカップラ２２では、パケット受信バッファ３７で、他のＰＭから転送されてきたデータを受け取り、パケット受信レジスタ３８に引き渡す。パケット受信レジスタ３８では、受信したデータをデコードし、指定された分散共有メモリアドレスロケーションにデータを書き込む。このようにして、送信側分散共有メモリから受信側分散共有メモリにコピーが行われる。なお、送信側分散共有メモリへの書き込みアクセスでは、受信側分散共有メモリへのコピーアクセスの完了を待たずにアクセスを完了する。



【００２８】本実施例においては、あるＰＭから他のＰＭにポイントツーポイントでコピーを行う構成を基本として説明しているが、ポイントツーマルチポイントでコピーを行うことも可能である。その場合には、モジュールＩＤディレクトリ６０には１つの分散共有メモリページアドレス６３に対して、複数の宛先ＰＭ　ＩＤ６４を登録しておく。その結果、分散共有メモリへの１回の書き込みに対して複数のパケットが生成され、それぞれ指定されたＰＭにパケットが送信されるので、複数のＰＭの分散共有メモリにコピーを行うことができる。また、同一の分散共有メモリページアドレス６３に対して、送信側ＰＭのモジュールＩＤディレクトリ６０には、受信側ＰＭ　ＩＤを登録し、受信側ＰＭのモジュールＩＤディレクトリ６０には、送信側ＰＭ　ＩＤを登録しておくことにより、分散共有メモリのエリアに送信側、受信側のどちらから書き込んでも、互いに相手側の分散共有メモリにコピーを行う、つまり双方向通信を行うことができる。

【００２９】図９は、図１におけるパケット送信レジスタおよびパケット受信レジスタに共通のデータフォーマット図である。ここでは、分散共有メモリアクセスタイプと割込みデータタイプの２種類のフォーマットを示している。図９において、８０，９０は属性識別フィールドであり、‘０’ならば分散共有メモリアクセスタイプであることを示し、‘１’ならば割込みデータタイプであることを示す。分散共有メモリアクセスタイプの場合には、モジュールＩＤ管理部３４が出力する『宛先ＰＭ　ＩＤ８１』，『分散共有メモリアドレス８２』，『書込みデータ８３』，『書込み単位８４』の各フィールドを含む。『書込み単位８４』フィールドは、書込みのデータ幅、例えばバイト、ハーフワード（１６ビット）、ロングワード（３２ビット）等を指定するものである。一方、割込みデータタイプの場合には、モジュールＩＤ管理部３４が出力する『宛先ＰＭ　ＩＤ９１』，割込み発生源を示す『送信モジュールＩＤ９２』，割込みの種別を示す『割込み識別データ９３』，『書込み単位９４』の各フィールドを含む。なお、宛先ＰＭ　ＩＤ８１，９１がある特定の値の場合には、放送型パケットとして全てのＰＭでパケットを受信するように構成することが可能である。

【００３０】図１０、図１１、図１２は、本発明において使用される各種のメッセージ通信のパターンと、その処理の流れを示す説明図である。図１０はシステム内メッセージ通信の場合である。図１０において、１２０は送信オブジェクト、１２１は受信オブジェクト、１２２はメッセージキュー、１２３は送信オブジェクト、１２

４は受信ディスクリプタリング、１２５はＩＯＣ配信オブジェクト、１２６はメッセージキュー、１２７は受信オブジェクトである。メッセージパターンは、以下のように分類される。

（ａ）システム内メッセージ通信・・・送信オブジェクトと受信オブジェクトが同一マルチプロセッサシステム（図１の１７）内に存在する場合である。システム内メッセージ通信には、次の２通りのケースが存在する。

（ａ１）ＰＭ内通信・・送信オブジェクト１２０と受信オブジェクト１２１が同一ＰＭ内に存在する場合である。ＰＭ内通信の場合、カーネルは受信オブジェクト１２１のメッセージキュー１２２のアドレスを知っているので、カーネルは、送信オブジェクト１２０から受信オブジェクト１２１のメッセージキュー１２２に直接、メッセージを登録することができる。受信オブジェクト１２７はメッセージキュー１２６からメッセージを取り出し、対応する処理を行う。

（ａ２）ＰＭ間通信・・送信オブジェクト１２３と受信オブジェクト１２７が同一マルチプロセッサシステム１７内の異なるＰＭに存在する場合である。ＰＭ間通信の場合、送信オブジェクト１２３の存在するＰＭ（送信ＰＭ）のカーネルからは、受信ＰＭ内の受信オブジェクト１２７の制御情報（例えばメッセージキュー１２６のアドレス）が見えないので、直接、受信オブジェクト１２７のメッセージキュー１２６にメッセージを登録することができない。そこで、送信ＰＭでは、送信カーネルが、事前に知っている受信ディスクリプタリング１２４にメッセージを登録し、受信ＰＭではＩＯＣ（Interobject Communication)配信オブジェクト１２５が、受信ディスクリプタリング１２４を読出し、受信オブジェクト１２７のメッセージキュー１２６にメッセージを登録する。

【００３１】図１１は、システム間メッセージ通信の場合である。図１１において、１４０は送信オブジェクト、１４１は送信ディスクリプタリング、１４２はＩＮＣ送信処理オブジェクト、１４３はアダプタ送信ディスクリプタリング、１４４はアダプタ受信ディスクリプタリング、１４６はメッセージキュー、１４７は受信オブジェクト、１５０は送信オブジェクト、１５１は送信ディスクリプタリング、１５２はＩＮＣ送信処理オブジェクト、１５３はアダプタ送信ディスクリプタリング、１５４はアダプタ受信ディスクリプタリング、１５５はＩＮＣ受信処理オブジェクト、１５６は受信ディスクリプタリング、１５７はＩＯＣ配信オブジェクト、１５８はメッセージキュー、１５９は受信オブジェクトである。

（ｂ）システム間メッセージ通信・・送信オブジェクトと受信オブジェクトが異なるマルチプロセッサシステム１７に存在する場合である。送信システムあるいは受信システムで中継ＰＭがない場合と、ある場合の２通りがある。



（ｂ１）中継ＰＭがない場合・・送信システムの場合、送信オブジェクト１４０の存在するＰＭ（送信ＰＭ）と、受信システム宛の通信リンクを持っているＰＭ（送信中継ＰＭ）とが一致している場合である。受信側システムの場合、通信リンクを終端しているＰＭ（受信中継ＰＭ）と受信オブジェクト１４７の存在するＰＭ（受信ＰＭ）とが一致している場合である。送信システムで送信中継ＰＭがない場合、送信ＰＭでは、カーネルが送信オブジェクト１４０からのメッセージを自ＰＭ内の送信ディスクリプタリング１４１に登録する。ＩＮＣ（Inter-System Communication)送信処理オブジェクト１４２は、送信ディスクリプタリング１４１を読み出してネットワークプロトコル処理を行い、アダプタ送信ディスクリプタリング１４３にメッセージを登録する。ネットワークアダプタ２４は、アダプタ送信ディスクリプタリング１４３からメッセージを取り出して、通信ネットワーク２６に送出する。

【００３２】一方、受信システムで受信中継ＰＭがない場合、通信ネットワーク２６からメッセージが送り届けられ、受信ＰＭのネットワークアダプタ２４がメッセージをアダプタ受信ディスクリプタリング１４４に登録する。そして、ＩＮＣ受信処理オブジェクト１４５がアダプタ用受信ディスクリプタリング１４４を読み出して、同一ＰＭ内の受信オブジェクト１４７のメッセージキュー１４６にメッセージを登録する。

（ｂ２）中継ＰＭがある場合・・送信システムの場合、送信オブジェクト１５０の存在するＰＭ（送信ＰＭ）が、受信システムへの通信リンクを持ち合わせておらず、他のＰＭ（送信中継ＰＭ）が持ち合わせている場合である。また、受信システムの場合、通信リンクを終端しているＰＭ（受信中継ＰＭ）と受信オブジェクト１５９の存在するＰＭ（受信ＰＭ）とが異なる場合である。送信システムでは、送信中継ＰＭがある場合、送信ＰＭのカーネルの送信処理プログラム１７が送信オブジェクト１５０からのメッセージを送信中継ＰＭ宛の送信ディスクリプタリング１５１に登録する。送信中継ＰＭのＩＮＣ送信処理オブジェクト１５２は送信ディスクリプタリング１５１を読み出して、ネットワークプロトコル処理を行ってアダプタ送信ディスクリプタリング１５３にメッセージを登録する。ネットワークアダプタ２４はアダプタ送信ディスクリプタリング１５３からメッセージを取り出して通信ネットワーク２６に送出する。

【００３３】また、受信システムで受信中継ＰＭがある場合、通信ネットワーク２６からメッセージが送り届けられ、受信ＰＭのネットワークアダプタ２４がメッセージをアダプタ受信ディスクリプタリング１５４に登録する。次に、受信中継ＰＭのＩＮＣ受信処理オブジェクト１５５がアダプタ受信ディスクリプタリング１５４を読み出し、最終宛先の受信ＰＭ宛の受信ディスクリプタリング１５６にメッセージを登録し、受信ＰＭではＩＯＣ

（Interobject Communication)配信オブジェクト１５７が受信ディスクリプタリング１５６を読み出し、受信オブジェクト１５９のメッセージキュー１５８にメッセージを登録する。なお、図１１のシステム間メッセージ通信の組み合わせパターンの数は、〔送信中継あり／なし〕，〔受信中継あり／なし〕の合計４通りである。図１２は、各アプリケーションオブジェクトに付与されたオブジェクトＩＤのフォーマットを示す図である。図１２において、１７０はオブジェクトＩＤ、１７１はそのオブジェクトが存在するマルチプロセッサシステムの番号で、通信ネットワーク内で一意に識別可能なように形成される。１７２はプロセッサモジュールＩＤ（ＰＭ　ＩＤ）で、マルチプロセッサシステム内で一意に識別可能の値をとる。１７３はローカルＩＤであり、ＰＭ内でオブジェクトを一意に識別するための番号である。

【００３４】図１３は、本発明における送信ディスクリプタと送信処理要求のメモリ配置例を示す図であり、図１４は、送信処理要求のフォーマット図である。ＰＭ＃１（１８－１）からＰＭ＃３（１８－３）宛の送信ディスクリプタリングを（ＸＤ１３－Ｓ，ＺＤ１３－Ｓ）のペアで、ＰＭ＃２（１８－２）からＰＭ＃３（１８－３）宛の送信ディスクリプタリングを（ＹＤ２３－Ｓ，ＺＤ２３－Ｓ）のペアで、ＰＭ＃３内の送信ディスクリプタリングをＺＤ３３－Ｓで、それぞれ表わす。また、ＰＭ＃１（１８－１）上のＰＭ＃３（１８－３）宛の送信処理ＦＩＦＯをＸＦ３－Ｓで、ＰＭ＃２（１８－２）上のＰＭ＃３（１８－３）宛の送信処理ＦＩＦＯをＹＦ３－Ｓで、ＰＭ＃３（１８－３）内の送信処理ＦＩＦＯをＺＦ３－Ｓで、それぞれ表わす。このように、図１３はシステム間ＰＭ中継通信の送信システムの例を示している。送信オブジェクトはＰＭ＃１（１８－１）、ＰＭ＃２（１８－２）、ＰＭ＃３（１８－３）に分散しており、送信中継ＰＭはＰＭ＃３（１８－３）とする。各ＰＭの送信オブジェクトから合計５個のメッセージＭｉ（ｉ＝１，２，・・５）がＰＭ＃３（１８－３）を経由して通信ネットワークに送出されるものとし、各メッセージの送信ディスクリプタを順にＱ１，Ｑ２，Ｑ３，Ｑ４，Ｑ５と表わし、それらに対応する送信処理要求を順にＱＡ１，ＱＡ２，ＱＡ３，ＱＡ４，ＱＡ５と表わす。送信ディスクリプタの構造は、後述の図１５（ｃ）に示されている。送信処理要求のデータ構造（フォーマット）は、図１４に示すように、要求元プロセッサモジュールＩＤ（ＰＭ　ＩＤ）１７５と、送信ディスクリプタへのポインタ１７６から構成されている。送信要求は、次のように伝達される。先ず、ＰＭ＃１（１８－１）でメッセージＭ１の送信要求が発生すると、ＰＭ＃１（１８－１）ではＰＭ＃３（１８－３）宛送信ディスクリプタリングＸＤ１３－Ｓに送信ディスクリプタＱ１を登録し、次にＰＭ＃３宛送信処理ＦＩＦＯエリアＸＦ３－Ｓに送信処理要求ＱＡ１を登録する。これらが分散メモリ



カップラ２２により、それぞれＰＭ＃３上の送信ディスクリプタリングＺＤ１３－ＳとＰＭ＃３上の送信処理要求ＦＩＦＯエリアＺＦ３－Ｓにコピーされる。

【００３５】次に、２番目のメッセージＭ２の送信要求がＰＭ＃３（１８－３）内で発生したと仮定すると、ＰＭ＃３（１８－３）内の送信ＭＢディスクリプタリングＺＤ３３－Ｓに送信ディスクリプタＱ２を、ＰＭ＃３（１８－３）宛送信処理ＦＩＦＯエリアＺＦ３－Ｓに送信処理要求ＱＡ２を登録する。この場合には、ＰＭ内通信であるため、他のＰＭへのコピーは行われない。なお、送信処理ＦＩＦＯエリアＺＦ３－ＳはＦＩＦＯメモリ構造であるため、最初に登録されたＱＡ１が保存されたままＱＡ２が追加登録される。次に、３番目のメッセージＭ３の送信要求がＰＭ＃２（１８－２）で発生したものとすると、ＰＭ＃２（１８－２）上のＰＭ＃３宛送信ＭＢディスクリプタリングＹＤ２３－Ｓに送信ディスクリプタＱ３がＰＭ＃３宛送信処理ＦＩＦＯエリアＹＦ３にＱＡ３が登録される。それらが、それぞれＰＭ＃３（１８－３）上の送信ＭＢディスクリプタリングＺＤ２３－ＳとＰＭ＃３上の送信処理要求ＦＩＦＯエリアＺＦ３－Ｓにコピーされる。送信処理ＦＩＦＯエリアＺＦ３－ＳにはＱＡ１，ＱＡ２が保存されたまま、ＱＡ３が追加登録される。送信ディスクリプタＱ４，Ｑ５についても、全く同じように、Ｑ５までの要求登録が終了した段階では、図１３のような記憶状態となる。

【００３６】この段階において、ＰＭ＃３（１８－３）のＩＮＣ送信処理オブジェクトは、ＰＭ＃３（１８－３）宛の送信処理ＦＩＦＯエリアＺＦ３－Ｓを読み出すと、入力した順に送信処理要求ＱＡ１，ＱＡ２，ＱＡ３，ＱＡ４，ＱＡ５を取り出すことができる。要求の中に記載されている送信元ＰＭ　ＩＤ１７５と送信ディスクリプタポインタ１７６から、対応する送信ディスクリプタＱｉの所在位置を求め、送信ディスクリプタＱｉからメッセージバッファの所在位置を求めることができる。このようにして、メッセージＭｉを順に処理してネットワークに送出していく。ここで、従来のポーリング方式では、３つの送信ディスクリプタリングＺＤ１３－Ｓ、ＺＤ２３－Ｓ、ＺＤ３３－Ｓを順にポーリングして送信要求の有無をチェックしており、処理要求の有無にかかわらず、全ての送信ディスクリプタリングを見る必要があった。そのため、読み出しても送信要求が見つからない空振りが生じることがあり、その結果、処理のオーバヘッドが大きかった。これに対して本実施例では、ＰＭ＃３（１８－３）で自ＰＭ宛送信処理ＦＩＦＯエリアＺＦ３－Ｓを周期的に読み出すだけで発生した送信要求を取り出すことができ、無駄な空振りが生じることがないので、極めて処理効率が高い。また、ＰＭ＃１（１８－１）から送信処理ＦＩＦＯエリアＸＦ３－Ｓを経由した送信処理ＦＩＦＯエリアＺＦ３－Ｓへの登録と、ＰＭ＃２（１８－２）から送信処理ＦＩＦＯエリアＹＦ３

－Ｓを経由した送信処理ＦＩＦＯエリアＺＦ３－Ｓへの登録が同時に発生しても、送信処理ＦＩＦＯエリアＺＦ３－ＳのＦＩＦＯメモリの構造により、登録要求が直列化されて全て蓄積されていくため、マルチプロセッサ間の競合処理を行う必要がなく、処理オーバヘッドは極めて軽減される。なお、図１３では、送信要求の場合を例に説明しているが、受信要求の場合にも全く同じように、自ＰＭ宛の受信要求を指定された受信処理ＦＩＦＯエリアから読み出すため、発生した受信要求のみを効率よく検出できる。

【００３７】図１５および図１６は、本発明で使用される各種データ構造を示す図である。図１５（ａ）はメッセージバッファ（ＭＢ）１８０のデータ構造であり、ＮＥＸＴＭＰ１８１は受信オブジェクト宛のメッセージをリスト構造で接続するためのポインタ、ＳＩＤ１８２はこのメッセージの送信元である送信オブジェクトのＩＤ、ＲＩＤ１８３はこのメッセージの宛先である受信オブジェクトのＩＤ、ＳＩＺＥ１８４はメッセージ本体（ＢＯＤＹ）１８６のサイズを表わしている。また、ＡＴＴＲ１８５はこのメッセージの付属属性を表示したフィールド、ＢＯＤＹ１８６はメッセージの中味を示すフィールドである。ＭＢ１８０の先頭アドレスは、ＭＢＡ２０２として付与される。各ＭＢは固定長であり、前述のように宛先ＰＭ　ＩＤが同じであるＭＢ群が分散共有メモリ２１の連続するエリアに割り付けられて１つのＭＢプールを構成している。各ＭＢプール対応にＭＢ管理マップが存在する。このＭＢ管理マップは、ＭＢプール内のＭＢ数に応じたＭＢ管理マップエントリから構成される。

【００３８】図１５（ｂ）はＭＢ管理マップエントリＶ１９０の構造を示しており、Ｖ＝０の場合には、対応するＭＢが『未使用』の状態であり、Ｖ＝１の場合には、対応するＭＢが『使用中』の状態を表わしている。ＭＢが１つ与えられると、その先頭アドレスＭＢＡ２００の値から、対応するＭＢ管理マップエントリＶ１９０のアドレスが簡単に計算で求められるように構成される。同じように、各宛先ＰＭ対応ＭＢプール毎に送信ディスクリプタリングと受信ディスクリプタリングが１つずつ存在して、両者を合わせてディスクリプタリングと呼ぶ。図１４で説明したように、送信ディスクリプタリングは他のシステムへのメッセージ制御情報を記憶し、受信ディスクリプタリングは他のシステムから、あるいはシステム内の他ＰＭからのメッセージ制御情報を記憶する。ディスクリプタリングの各エントリ（ディスクリプタ）はＭＢの各々に対応している。図１５（ｃ）は、ディスクリプタ２００の構造を示したものであって、２０１は論理リンク番号で、対応するＭＢ上のメッセージを通信ネットワーク２６経由で他のシステムに転送する場合に使用する通信リンクの識別番号である。２０２は対応するＭＢの先頭アドレスＭＢＡを記憶するフィールドで、



図１５（ａ）で示したＭＢＡ２０２と同一である。２０３は実行権を表わす情報であり、実行権＝『ＥＮＱ』であるならば、ディスクリプタリングにディスクリプタを登録できる状態（エンキュー）におることを示しており、実行権＝『ＤＥＱ』であれば、ディスクリプタリングからそのディスクリプタを取り出せる状態（デキュー）にあることを示している。

【００３９】図１５（ｄ）はアプリケーションオブジェクトの実行に必要な各種の制御データを記憶するオブジェクトコントロールブロック（ＯＣＢ）の構造を示したもので、ＯＣＢＡ２１１はＯＣＢの先頭アドレスを示すポインタ、ＮＥＸＴＯＰ２１２はＯＣＢをレディキューにリンクする時のポインタ、ＭＳＧＰ２１３はオブジェクト宛のメッセージを記憶しているＭＢのアドレスを示す。ＳＴＡＴＵＳ２１４は、オブジェクトの実行状態を示すフィールドであり、送受信オブジェクト間の同期制御に使用される。その他の制御２１５は、その他の制御フィールドである。図１６では、プロセッサ１９実行待ちのＯＣＢを登録したレディキューの構造と、各種制御データ構造との関係を示している。ＲＥＡＤＹＰ２２０で示されるレディキューに２つのＯＣＢ（ＯＣＢｉ２２１，ＯＣＢｊ２２２）が登録されており、ＯＣＢｉ２２１は２個のメッセージを受信し、ＯＣＢｊ２２２は３個のメッセージを受信している状態にある例を示している。

【００４０】図１７は、システムルーチングテーブルとＰＭルーチングテーブルの構造図である。受信オブジェクトＩＤ（ＲＩＤ）（図１５の１８３）の値を基にして、メッセージをシステム内のどのＰＭを経由して送受信するかを指定するルーチングテーブルであって、本実施例では、システムルーチングテーブル２４９とＰＭルーチングテーブル２８９から構成され、それぞれ各ＰＭのローカルメモリ２０に記憶されている。システムルーチングテーブル２４９の各エントリは、システムＩＤフィールド２５０、システム通信モードフィールド２５１、通信リンクフィールド２５２、送信中継ＰＭフィールド２５３から構成されている。システムＩＤフィールド２５０に『自システムＩＤ２６０』が登録されているエントリの場合には、そのエントリのシステム通信モードフィールド２５１は、『システム内通信２６１』と表示されている。一方、システムＩＤフィールド２５０に『他システムＩＤ２７０』が登録されているエントリの場合には、そのエントリのシステム通信モードフィールド２５１は『システム間通信２７１』と表示され、通信リンクフィールド２５２には、他システムに結合されている通信リンクの『通信リンクＩＤ２７２』が登録されている。また、送信中継ＰＭフィールド２５３には、通信リンクＩＤ２７２に対応する通信リンクを収容している『送信中継ＰＭのＩＤ２７３』を記憶している。あるマルチプロセッサシステム１７が、他のＫ個のマルチプ

ロセッサシステムと接続されている場合には、システムルーチングテーブル２４９のエントリ総数は（１＋K）個となる。

【００４１】PMルーチングテーブル２８９は、PM IDフィールド２９０、PM通信モードフィールド２９１、MB管理マップアドレスフィールド２９２から構成される。PM IDフィールド２９０に『自PM ID３００』が登録されているエントリの場合には、そのエントリのPM通信モードフィールド２９１には『PM内通信３０１』と記述されており、MB管理マップアドレスフィールド２９２には『PM内通信MB管理マップベースアドレス３０２』が登録されている。また、PM IDフィールド２９０に、『他PM ID３１０』が登録されているエントリの場合には、PM通信モードフィールド２９１には『PM間通信３１１』と記述されており、MB管理マップアドレスフィールド２９２には、そのPMIDに対応する『PM間通信MB管理マップベースアドレス３１２』が登録されている。各フィールドの詳細な使用方法は後述する。なお、あるマルチプロセッサシステム１７がJ個のPMから構成されている場合には、PMルーチングテーブルのエントリ総数はJ個である。

【００４２】図１８は、本発明におけるシステム内PM内通信のタイムチャートであり、図１９は同じくシステム内PM間通信のタイムチャートであり、図２０はシステム間PM無中継通信のタイムチャートであり、図２１はシステム間PM中継通信のタイムチャートである。なお、図１８～図２１のプロセッサの実行ステップのうち、太い罫線で示されたステップ（例えば、図１８のステップS１００）はプロセッサのスーパバイザモードで実行している部分であり、これに対して細い罫線で示されたステップ（例えば、図１８のステップS１０２）はプロセッサのユーザモードで実行している部分である。図２２～図３２は、アプリケーション（APL）オブジェクトからの要求により、カーネル自身のプログラムあるいはカーネルが起動するプログラムのフローチャートであって、いずれも全てプロセッサのスーパバイザモードで実行される。図２２は、カーネルの一部であって、アプリケーション（APL）オブジェクトの起動および終了処理を行うプログラムのフローチャートである。また、図２３および図２４は、MB捕捉処理プログラムおよびMB解放処理プログラムの各フローチャートであって、APLオブジェクトからの要求により起動される。また、図２５および図２６は、送信処理および受信処理プログラムのフローチャートであり、これらもAPLオブジェクトからの要求により起動される。

【００４３】図２７および図２８は、送信ディスクリプタリング、受信ディスクリプタリング、アダプタ送信ディスクリプタリング、アダプタ受信ディスクリプタリングに共通なエンキュー処理（各ディスクリプタリングにエントリを登録する処理）およびデキュー処理（各ディスクリプタリングからエントリを削除する処理）のプログラムのフローチャートである。図２７（a）はディスクリプタリングの一般的な論理構造を示す図、図２７（b）はエンキュー処理のフローチャート、図２８はデキュー処理のフローチャートである。図２７（a）において、４００はリードポインタ（RP）であり、ディスクリプタリングからエントリを取り出す場合（デキュー処理）のエントリ読み出しアドレスを指定する。４０１はライトポインタ（WP）であり、ディスクリプタリングにエントリを登録する場合（エンキュー処理）のエントリ書き込みアドレスを指定する。各エントリは実行権フィールド４０３と、処理要求内容を記憶するディスクリプタアイテムフィールド４０４に分割される。ディスクリプタアイテムフィールド４０４は、ディスクリプタリングの種類（例えば、送信ディスクリプタリングか／アダプタ送信ディスクリプタリングかの違い）に依存して内容が変わることがある。実行権フィールド４０３は、そのエントリにアクセスする主体がエンキュー側にあるか／デキュー側にあるかを指定するもので、『ENQ』と表わされているときには新しいエントリを登録してもよいことを意味し、『DEQ』の表わされているときには登録されているエントリを取り出してもよいことを意味する。エンキュー側はエントリを登録する毎にWP４０１をインクリメントし、次のエンキュー位置を知らせる。また、デキュー側はエントリを取り出す毎にRP４００をインクリメントし、次のデキュー位置を知らせる。

【００４４】図２７にエンキュー処理，図２８にデキュー処理の各フローが示される。エンキュー処理では、ステップ４２０で、WP４０１で示されるエントリの実行権フィールド４０３が『ENQ』と表示されているか否かをテストする。もし、ENQと表示されているならば、新しいエントリを登録してもよいことを意味するので、ステップ４２１でそのディスクリプタアイテムフィールド４０４に新エントリ情報を書き込み、ステップ４２２でそのエントリの実行権フィールド４０３を『DEQ』に書き替える。ステップ４２３では、このエントリの登録によりディスクリプタリングの最後に到達したか否かをテストする。到達していなければ、ステップ４２４でWP４０１をインクリメントし、次のエンキュー位置を設定する。ディスクリプタリングの最後に到達していれば、ステップ４２５でWP４０１の値をディスクリプタリングのベースアドレスにセットすることにより、ディスクリプタリングをサイクリックに使用する。ステップ４２０で実行権フィールド＝『DEQ』と表示されているときには、これはディスクリプタリング４０２に前回登録したエントリがデキュー側により未だ読み出されずに残っていることを示すので、ステップ４２６でディスクリプタリングオーバフローのエラー処理を実行す

る。

【００４５】一方、デキュー処理では、ステップ４３０でＲＰ４００で示されるエントリの実行権フィールド４０３が『ＤＥＱ』と表示されているか否かをテストする。もし、ＤＥＱと表示されていれば、エントリが登録されており、そのエントリを取り出してもよいことを意味するので、ステップ４３１でそのディスクリプタリングアイテムフィールド４０４からディスクリプタを取り出し、ステップ４３２でその実行権フィールド４０３を『ＥＮＱ』に書き替える。ステップ４３３では、このエントリがディスクリプタリングの最後に登録されていたものか否かをテストする。ディスクリプタリング４０２の最後でなければ、ステップ４３４でＲＰ４００をインクリメントし、次のデキュー位置を知らせて、ステップ４３５で取り出したディスクリプタエントリを呼び出し、元に渡す。ディスクリプタリングの最後であれば、ステップ４３６でＲＰ４００の値をディスクリプタリングのベースアドレスにセットすることにより、ディスクリプタをサイクリックに使用する。ステップ４３０で実行権フィールド＝『ＥＮＱ』と表示されていれば、これはディスクリプタリングで前回のサイクルエントリを削除してから、エンキュー側により未だ新しくエントリが登録されていないことを示しているので、何もせず、リターンする。

【００４６】図２９は、ＩＯＣ配信処理プログラムのフローチャートである。ＩＯＣ配信プログラムは、定期的に自ＰＭ宛の受信処理ＦＩＦＯ通信エリア（図５～図７参照）を読み出し、他のＰＭから受信処理要求が送られてきたか否かをチェックする。図３０は、ＩＮＣ送信処理プログラムのフローチャートである。また、図３１および図３２は、ＩＮＣ受信処理プログラムのフローチャートである。以下、図１８～図３３を参照しながら、実施例の動作を説明する。図１０および図１１に示した分類に従って、次の４つのメッセージ通信のケースについて詳述する。

（ケース１）システム内ＰＭ内通信

（ケース２）システム内ＰＭ間通信

（ケース３）システム間中継ＰＭなしの通信（システム間ＰＭ無中継通信）

（ケース４）システム間中継ＰＭありの通信（システム間ＰＭ中継通信）

【００４７】（ケース１）システム内ＰＭ内通信

（ａ）送信側の処理

プロセッサモジュール（ＰＭ）１８－１内の送信オブジェクトから同じＰＭ（１８－１）内の受信オブジェクトにメッセージを転送する場合を考える。送信オブジェクトと受信オブジェクトは互いに非同期で実行されるので、その実行順序は種々のケースが考えられるが、ここでは簡単のために送信オブジェクトが受信オブジェクトよりも先に実行される図１８のケースを仮定する。図１



８において、Ｓ１００はＡＰＬオブジェクト起動、Ｓ１０１はＭＢ捕捉、Ｓ１０２はメッセージ書込み、Ｓ１０３は送信処理、Ｓ１０４はＡＰＬオブジェクト終了である。また、Ｓ１１０はＡＰＬオブジェクト起動、Ｓ１１１は受信処理、Ｓ１１２はメッセージ読出し、Ｓ１１３はＭＢ解放、Ｓ１１４はＡＰＬオブジェクト終了である。

「ステップＳ１００」カーネルは、ステップＳ１００で、アプリケーション（ＡＰＬ）オブジェクトを実行させる準備を行う。ＡＰＬオブジェクトとは、ユーザモードで動作するアプリケーションプログラムのことである。ステップＳ１００の『ＡＰＬオブジェクト起動』の内容は、図２２に示すように、ステップ３３０でレディキュー（実行可能なオブジェクトコントロールブロック（ＯＣＢ）（図１６の２１０）がリスト構造でリンクされたもの（図１６参照）から、次に実行すべきＡＰＬオブジェクトのＯＣＢ２１０を取り出す。次に、ステップ３３１で、ＡＰＬオブジェクトに与えられているオブジェクトＩＤ（図１２の１７０）を分散メモリプロテクタ２３内のカレントコントロールレジスタ（ＣＣＲ）５１に設定する。このタイミングは、図１８のタイミングｔ１００により示される。次に、図２２のステップ３３２で、必要な情報をＯＣＢ２１０からプロセッサ１９にロードして、ＡＰＬオブジェクトの実行を開始する。ここでは、ＡＰＬオブジェクトが送信オブジェクトであるため、送信オブジェクトの実行が開始される。

【００４８】「ステップＳ１０１」送信オブジェクトが、ＭＢの捕捉をカーネルに要求すると、カーネルは図２３のＭＢ捕捉処理プログラムを実行する。図２３のステップ３５０で、メッセージを受信する受信オブジェクトのＩＤ（図１２のＲＩＤ１７０）内のシステムＩＤフィールド１７１を取り出す。この値をシステムルーチングテーブル（図１７の２４９）のシステムＩＤフィールド２５０の各エントリと比較し、一致したエントリのシステム通信モードフィールド２５１、通信リンクフィールド２５２、送信中継ＰＭフィールド２５３の値を取り出す。ここでは、同一システム内の同一ＰＭ内通信であるため、『自システムＩＤ』と記されたエントリ２６０と一致し、そのシステム通信モードフィールド２５１は、『システム内通信』２６１と表示されているため、ステップ３５１からステップ３５２に進む。ステップ３５３では、受信オブジェクトＩＤ（ＲＩＤ）のＰＭ　ＩＤフィールド１７２をＰＭルーチングテーブル（図１７の２８９）のＰＭ　ＩＤフィールド２９０の各エントリと比較して、一致したエントリのＰＭ通信モードフィールド２９１と、ＭＢ管理マップアドレスフィールド２９２の値を取り出す。ここでは、同一システム内の同一ＰＭ内通信であるため、『自ＰＭ　ＩＤ』と記述されたエントリ３００と一致し、その結果、『ＰＭ内通信３０１』と表示されたＰＭ通信モードフィールドと、『ＰＭ

内通信ＭＢ管理マップベースアドレス３０２』と表示されたＭＢ管理マップアドレスフィールドとを取り出す。

【００４９】ステップ３５３では、選択したＭＢ管理マップアドレス２９２で指定されたＭＢ管理マップを検索し、空きを表示しているＭＢ管理マップエントリを検出し、それに対応する空きＭＢアドレスＭＢＡ（図１５の２０２）を求め、そのＭＢ管理マップエントリ（図１５の１９０）を『使用中』に書き替える。ＭＢ管理マップエントリとＭＢとは、１対１に対応するように構成されており、ＭＢ管理マップエントリアドレスからＭＢアドレスを求めることができるとともに、逆にＭＢアドレスからもＭＢ管理マップエントリアドレスを求めることができる。ここでは、ＰＭ内通信用ＭＢ管理マップが選択され、その中の適当な空きＭＢ、例えば図５，図７のＭＢ（Ｘ１１－２）が選択される。ステップ３５３の最後で、ＭＢ（Ｘ１１－２）に対応するＭＢ管理マップエントリ（ＸＭ１１－２）が図１７の及楽石械胤ｔ１０１で『使用中』に書き替えられる。続いて、ステップ３５４で、ＭＢ（Ｘ１１－２）に対応するメモリケーパビリティレジスタ（ＭＣＲ）５３のアドレスを求め、ＭＣＲ５３に、このＭＢへのアクセス権を獲得したオブジェクト、つまり送信オブジェクトのＩＤ（ＳＩＤ）が図１８のタイミングｔ１０２で登録される。以上が、図１８のステップＳ１０１のＭＢ捕捉処理である。

【００５０】「ステップＳ１０２」送信オブジェクトは、図１８のステップＳ１０２で、作成したメッセージをＭＢ（Ｘ１１－２）にタイミングｔ１０３で書き込む。図１８のプロテクションウィンドウ開放期間ｗ１は、ＣＣＲ５１とＭＣＲ５３の両方に値が設定されている区間で、ＭＣＲとＣＣＲとが指定しているＡＰＬオブジェクト（送信オブジェクト）は、ＭＢ（Ｘ１１－２）にアクセスすることができるが、他のＡＰＬオブジェクトはアクセスすることができなくなり、ＭＢに対する強固な保護が実現される。

【００５１】「ステップＳ１０３」送信オブジェクトがステップＳ１０３でカーネルに送信処理を依頼すると、カーネルは、図２５の送信処理プログラムを実行する。先ず、図２５のステップ３８０でＭＢ（Ｘ１１－２）に対応するＭＣＲ５３を図１８のタイミングｔ１０４でクリアする。これで、図１８のプロテクションウィンドウ開放期間ｗ１が終了するため、以後、カーネル以外はＭＢ（Ｘ１１－２）にアクセスできなくなる。次に、図２５のステップ３８１で、通信モードをチェックする。ここでは、『システム内ＰＭ内通信』であるため、ステップ３８２で受信オブジェクトのＯＣＢ２１０のメッセージキューにＭＢ（Ｘ１１－２）のアドレス（ＭＢＡ）を登録する。続いて、ステップ３８３でＯＣＢ２１０のＳＴＡＴＵＳフィールド（図１５の２１４）が『送信側実行待ち』となっているか否かをテストする。『送信側実行待ち』は、受信オブジェクトが送信オブジェクトより



も先に実行された結果、受信オブジェクトが送信オブジェクトの実行を待っている状態を表わす。このケースでは、送信オブジェクトが受信オブジェクトよりも先に実行されるため、ステップ３８３は『ＮＯ』となって、ステップ３８５でＯＣＢ２１０のＳＴＡＴＵＳフィールド２１４を『受信側実行待ち』と設定して、受信オブジェクトの実行を待つ状態になったことを表示する。

【００５２】「ステップＳ１０４」ステップＳ１０４では、カーネルがＡＰＬオブジェクト終了処理を起動する。ＡＰＬオブジェクト終了処理プログラムは、図２２に示されており、ステップ３４０でカレントケーパビリティレジスタＣＣＲ５１の値を図１８のタイミングｔ１０５でクリアする。続いて、図２２のステップ３４１で送信オブジェクトの再開に必要な情報をＯＣＢ２１０に退避することにより、送信オブジェクトの処理を完了する。

【００５３】（ｂ）受信側の処理

「ステップＳ１１０」図１８のステップＳ１１０において、カーネルは次に実行すべきオブジェクトして受信オブジェクトを選択し、図２２のＡＰＬオブジェクト起動プログラムを実行する。送信オブジェクトの場合と同じように、図１８のタイミングｔ１１０で受信オブジェクトＩＤ（ＲＩＤ）（図１５の１８３）をカレントケーパビリティレジスタ（ＣＣＲ）５１に設定する。

【００５４】「ステップＳ１１１」受信オブジェクトの実行が開始され、受信オブジェクトがカーネルに受信処理を要求すると、カーネルは図２６の受信処理プログラムを起動する。図２６のステップ３９０では、受信オブジェクトのＯＣＢ２１０のＳＴＡＴＵＳフィールド２１４が『受信側実行待ち』または『同期完了』となっているか否かをテストする。『受信側実行待ち』は、送信オブジェクトが受信オブジェクトよりも先に実行された結果、受信オブジェクトの実行を待っている状態である。また、『同期完了』は、受信オブジェクトが送信オブジェクトよりも先に実行されたため、途中で休止し、その後、送信オブジェクトが実行されたことにより、送受信間の同期がとれた状態を示している。このケースでは、図１８のステップＳ１０３で、ＳＴＡＴＵＳフィールド２１４を『受信側実行待ち』と設定しているため、ステップ３９０は『ＹＥＳ』となる。これは、メッセージが送信側から既に届いているため、直ちにメッセージキューにつながっているメッセージをキューから外し、そのＭＢ（Ｘ１１－２）のアドレスを取り出す。続いて、ステップ３９２では、このＭＢアドレスに対応するＭＣＲ５３に受信オブジェクトＩＤ（ＲＩＤ）１８３を設定する。このタイミングは、図１８のタイミングｔ１１１で示される。これにより、ＣＣＲ５１とＭＣＲ５３の両方に受信オブジェクトＩＤ（ＲＩＤ）１８３が設定されたので、以後、図１８のプロテクションウィンドウ開放期間（ｗ２）中は、ＭＢ（Ｘ１１－２）が受信オブジェク

ト以外のＡＰＬオブジェクトからアクセスできないようにプロテクトされることになる。

【００５５】「ステップＳ１１２」受信オブジェクトが、ＭＢ（Ｘ１１－２）からタイミングｔ１１２でメッセージを読み出して、対応する処理を行う。

【００５６】「ステップＳ１１３」受信オブジェクトは、ステップＳ１１３でカーネルにＭＢ解放を依頼する。カーネルは、図２４のＭＢ解放処理プログラムを起動する。図２４において、ステップ３６０で、ＭＢ（Ｘ１１－２）に対応するＭＣＲ５３をクリアする。このタイミングは、図１８のタイミングｔ１１３に示されている。続いて、図２４のステップ３６１でＭＢ（Ｘ１１－２）に対応するＭＢ管理マップエントリ（ＸＭ１１－２）を『空き』に更新する。このタイミングは、図１８のタイミングｔ１１４に示されている。ＭＢ管理マップエントリを『空き』に更新すると、以後は、いつでもこのＭＢ（Ｘ１１－２）を再利用することが可能となる。

【００５７】「ステップＳ１１４」カーネルがＡＰＬオブジェクト終了処理を実行して、カレントケーパビリティレジスタ（ＣＣＲ）５１の値を図１８のタイミングｔ１１５でクリアする。ＡＰＬオブジェクト終了処理は、図２２に示されており、送信オブジェクトの場合と同じであるため説明を省略する。以上のように、システム内ＰＭ内通信の場合には、送信オブジェクトと受信オブジェクトがユーザレベルで直接アクセスできる分散共有メモリエリアに、メッセージバッファを確保することにより、メッセージバッファ間の無駄なコピーを避けることができるので、効率的なメッセージ通信が可能となる。

【００５８】（ケース２）システム内ＰＭ間通信

（ａ）送信ＰＭ側の処理

マルチプロセッサシステム１７－１内のプロセッサモジュール（ＰＭ）１８－１内の送信オブジェクトから、ＰＭ１８－２内の受信オブジェクトにメッセージを転送する場合を仮定する。さらに、図１９に示すように、送信オブジェクトの実行終了前に受信オブジェクトの実行が開始される場合を考える。使用するＭＢを図５のＸ１２－１とすると、対応するＭＢ管理マップエントリはＸＭ１２－１となる。また、この処理依頼内容の詳細は受信ディスクリプタリングＸＤ１２－Ｒに、受信処理要求は受信ＰＭ１８－２宛の受信処理ＦＩＦＯエリアＸＦ２－Ｒに登録されるものとする。処理の流れを、図１９の各ステップに従って説明する。Ｓ１２０はＡＰＬオブジェクト起動、Ｓ１２１はＭＢ捕捉、Ｓ１２２はメッセージ書込み、Ｓ１２３は送信処理、Ｓ１２４はＡＰＬオブジェクト終了である。また、Ｓ１１０はＡＰＬオブジェクト起動、Ｓ１１１は受信処理、Ｓ１１２はメッセージ読出し、Ｓ１１３はＭＢ解放、Ｓ１１４はＡＰＬオブジェクト終了である。

【００５９】「ステップＳ１２０」送信ＰＭ１８－１のカーネルは、ステップＳ１２０でＡＰＬオブジェクト起動プログラム（図２２）を実行し、送信オブジェクトＩＤ（ＳＩＤ）を図１９のタイミングｔ１２０でＰＭ１８－１のＣＣＲ５１－１に設定し、送信オブジェクトの実行を開始する。



【００６０】「ステップＳ１２１」送信オブジェクトはＭＢ捕捉をカーネルに要求し、カーネルは図２３のＭＢ捕捉処理プログラムを実行する。このケースでは、同一システム内で２つのＰＭにまたがった通信であるため、図２３のステップ３５０で検索して取り出されるシステム通信モードフィールド（図１７の２５１）の値は『システム内通信』２６１となり、ステップ３５１からステップ３５２に進む。ステップ３５２で検索して取り出されるＰＭ通信モードフィールド（図１７の２９１）の値は『ＰＭ間通信３１１』となり、これに対応して『ＰＭ間通信ＭＢ管理マップベースアドレス３２２』と表示されたＭＢ管理マップアドレスフィールドを取り出す。ステップ３５３では、選択した『ＰＭ間通信ＭＢ管理マップベースアドレス３２２』で指定されたＭＢ管理マップを検索し、空きを表示しているＭＢ管理マップエントリを検出して、それに対応する空きＭＢアドレスを求め、そのＭＢ管理マップエントリを『使用中』に書き替える。この例では、ＰＭ１８－１からＰＭ１８－２への通信用ＭＢＸ１２－１が図１９のタイミングｔ１２１で『使用中』に書き替えられるが、この時点で同時にＸＭ１２－１と同一アドレスを持つ受信側分散共有メモリ２１－２のＭＢ管理マップエントリＹＭ１２－１も書き替えられる。この書き替えは、次のような操作で行われる。

【００６１】先ず、送信ＰＭ１８－１のプロセッサ１９－１がＭＢ管理マップエントリＸＭ１２－１の書き替え命令を実行すると、ＭＢ管理マップエントリＸＭ１２－１のアドレスと書き替えデータがプロセッサバス３０－１（図１参照）に送出される。送信側分散メモリカップラ２２－１では、送信側分散共有メモリ２１－１への書き込みアクセスがあった場合に、バス信号デコーダ３１－１が分散共有メモリ２１－１への書き込みアクセスであることを検出し、信号線３２－１を経由して転送制御部３３－１を起動する。転送制御部３３－１は、ＭＢ管理マップエントリＸＭ１２－１のアドレスと書き替えデータを取り込み、モジュールＩＤ管理部３４－１に、ＭＢ管理マップエントリのページアドレスを供給する。モジュールＩＤ管理部３４－１のＣＡＭコマンドレジスタ（図８の６７－１）には、予め『キーレジスタ（図８の６６－１）との比較動作』を指示するコマンドが入れられているため、ＭＢ管理マップエントリＸＭ１２－１のページアドレスがキーレジスタ６６－１に格納されると、キーレジスタ（６６－１）とＣＡＭセル部（図８の６１－１）との一斉比較が行われる。ＣＡＭセル部６１－１の各エントリは、他のＰＭの分散共有メモリエリアと重複しているエリアのページアドレス６３を含んでい

る。

【００６２】本実施例においては、受信ＰＭ１８－２との間で共有されているＸＭ１２－１のページアドレスと、受信ＰＭ１８－２ＩＤのペアがそれぞれＣＡＭセル部６１－１とデータメモリ部６２－２に登録されている。従って、ＣＡＭセル部６１－１における比較動作により一致が検出され、対応する受信ＰＭ１８－２のＩＤがデータメモリ部６２－１から取り出される。これを、パケット送信レジスタ３５－１に転送する。パケット送信レジスタ３５－１では、図９に示すように、属性識別フィールド８０を‘０’に設定して、共有分散メモリアクセスタイプのパケットであることを表示し、宛先ＰＭ　ＩＤフィールド８１には、モジュールＩＤ管理部から取り出した値（ＰＭ１８－２のＩＤ）を設定する。また、分散共有メモリアドレスフィールド８２と書き込みデータフィールド８３には、プロセッサバス３０－１上のアドレス（ＸＭ１２－１のアドレス）とデータ（『使用中』表示データ）を載せ、書き込み単位フィールド８４には、プロセッサバス３０－１が指示する信号（バイト書き込みとバイト位置情報）を載せて送信パケットを編集する。これをパケット送信バッファ３６－１に送出すると、パケット送信バッファ３６－１は送信パケットをプロセッサモジュール間通信路２５に送出する。プロセッサモジュール間通信路２５は、送信パケットの宛先ＰＭ　ＩＤフィールド８１を見てルーチングを行い、受信ＰＭ１８－２のパケット受信バッファ３７－２に送り届ける。

【００６３】受信ＰＭ１８－２のパケット受信バッファ３７－２のデータはパケット受信レジスタ３８－２に入れられてパケットデコーダ３９－２でデコードされ、共有分散メモリ２１－２への書き込みであることがわかると、転送制御部３３－２を起動する。転送制御部３３－２は、パケット受信レジスタ３８－２内の分散共有メモリアドレス（ＸＭ１２－１のアドレス）、書き込みデータ（『使用中』表示データ）、書き込み単位の情報をプロセッサバス３０－２経由で分散共有メモリ２１－２に送ることにより、ＸＭ１２－１への書き込みデータと同じ値が受信側分散共有メモリ２１－２の同じアドレス位置、つまりＹＭ１２－１に書き込まれる。このようにして、分散共有メモリへの書き込み側が自分の分散共有メモリへのローカルな書き込みを行うだけで、分散メモリカップラ２２の働きにより自動的に宛先の分散共有メモリにも同じ値がコピーされるので、ＤＭＡのようなソフトウェアによるコピーオーバーヘッドを伴わないという利点がある。

【００６４】なお、コピーバック型のキャッシュメモリを備えるプロセッサ１９では、分散共有メモリ２１への書き込みアクセスがキャッシュヒットすると、その時点でキャッシュメモリへの書き込みだけが行われ、分散共有メモリ２１への書き込みは直ちに行われないことがある。その場合には、キャッシュメモリへの書き込み後、キャッシュメモリ内の書き込みデータを分散共有メモリに戻す命令を実行することにより、送信側分散共有メモリ２１－１への書き込みを確実に行うので、受信側分散共有メモリ２１－２へのコピーも確実に行うことができる。次に、図２３のステップ３５４では、ＭＢ（Ｘ１２－１）に対応するメモリケーパビリティレジスタ５３－１のアドレスを求め、ＭＣＲ５３－１に対しこのＭＢへのアクセス権を獲得した送信オブジェクトのＩＤ（ＳＩＤ）を図１９のタイミングｔ１２２で登録する。以上で、ステップＳ１２１のＭＢ捕捉処理を終了する。



【００６５】「ステップＳ１２２」送信オブジェクトが作成したメッセージを送信側分散共有メモリのＭＢ（Ｘ１２－１）にタイミングｔ１２３で書き込みを行う度毎に、この書き込みデータが分散メモリカップラ２２により受信側分散共有メモリの同一アドレスのＭＢ（Ｙ１２－１）にも伝搬して書き込まれる。その操作は、図１９のタイミングｔ１２１で説明した通りである。なお、本実施例のメッセージ書き込みステップでは、図９に示すように、書き込み単位、例えば４バイトの書き込みデータ毎に１つのパケットを構成し、分散メモリカップラ２２間で送受する構成であるがメッセージのようにメモリの連続エリアへの書き込みが行われる場合には、図９の送受信パケットを、先頭の書き込みアドレスに続いて複数個の書き込みデータを１パケットにまとめた構成にして、分散メモリカップラ２２間で送受する方法も可能である。その場合には、連続エラアへの書き込みが行われているのか、または単発エリアの書き込みが行われているのかを分散メモリカップラ２２側で事前に検知することができないため、分散メモリカップラ２２は、予め決められた時間内に決められた回数の連続アドレスエリア書き込みがあれば、それらを１パケットにまとめて送る。また、決められた時間内に決められた回数よる少ない連続アドレスエリア書き込みであれば、タイムアウトになった時点で、それまでにたまっている連続アドレスエリア書き込みデータを１パケットにまとめて送出すればよい。図１９において、連続番地ＣＣＲ５１－１とＭＣＲ５３－１の両方の値が設定されている区間（ステップＳ１２１からステップＳ１２４までの区間）では、ＭＣＲとＣＣＲにより指定された送信オブジェクトがＭＢ（Ｘ１２－１）にアクセスすることができるが、ＰＭ１８－１内の他のＡＰＬオブジェクトはアクセスすることができない。

【００６６】「ステップＳ１２３」送信オブジェクトは、カーネルに送信処理を依頼する。送信処理プログラムは、図２５に示されるように、先ずステップ３８０でＭＢ（Ｘ１２－１）に対応するＭＣＲ５３－１をクリアする。この動作は、図１９のタイミングｔ１２４に示されており、これによりプロテクションウィンドウ開放期間が終了するので、これ以後、カーネル以外はＭＢ（Ｘ

１２－１）にアクセスすることができなくなる。次に、図２５のステップ３８１で通信モードをチェックする。ここでは、『システム内ＰＭ間通信』であるため、ステップ３８６に進む。ステップ３８６では、図１９のステップＳ１２１で検索一致したＰＭ　ＩＤ（図１７の２９０）、つまりＰＭ１８－２宛の受信ディスクリプタリングＸＤ１２－Ｒに受信処理制御情報を登録（エンキュー）すると同時に、図１４のフォーマットの処理要求を受信ＰＭ１８－２宛の受信処理ＦＩＦＯ（ＸＦ２－Ｒ）に書き込む。これらの書き込みは、図１９のタイミングｔ１２５で行われる。受信ディスクリプタリングＸＤ１２－Ｒへのエンキュー処理は、図２７に示されているので、説明は省略する。受信ディスクリプタリングＸＤ１２－Ｒに書き込みが行われると、書き込みデータが分散メモリカップラ２２－１により受信ＰＭ１８－２にも転送され、その分散共有メモリ２１－２の同一アドレスであるＹＤ１２－Ｒにも書き込まれる。

【００６７】一方、受信処理ＦＩＦＯへの書き込みについて、送信ＰＭ１８－１側の受信処理ＦＩＦＯエリア（ＸＦ２－Ｒ）は、通常のＲＡＭで構成されているが、受信ＰＭ１８－２側の受信処理ＦＩＦＯエリア（ＹＦ２－Ｒ）はＦＩＦＯメモリで構成されている。従って、ＸＦ２－Ｒへの書き込みデータが分散メモリカップラ２２により受信側の分散共有メモリの受信処理ＦＩＦＯエリア（ＹＦ２－Ｒ）に伝達されると、ＹＦ２－Ｒでは、以前に書き込まれたデータを保存したまま、新しい書き込みデータが追加して書き込まれる。以上により、メッセージ送信に必要な情報が受信ＰＭ１８－２に全て伝達された。

【００６８】「ステップＳ１２４」カーネルが図２２のＡＰＬオブジェクト終了処理を起動し、カレントケーパビリティレジスタ（ＣＣＲ）５１－１の値を図１８のタイミングｔ１２６でクリアする。

【００６９】（ｂ）受信ＰＭ１８－２側の処理

「ステップＳ１３０」図１９のステップＳ１３０において、カーネルは次に実行すべきオブジェクトして受信オブジェクトを選択し、図１９のタイミングｔ１３０で受信オブジェクトＩＤ（ＲＩＤ）を受信ＰＭ１８－２内のカレントケーパビリティレジスタ（ＣＣＲ）５１－２に設定する。

【００７０】「ステップＳ１３１」受信オブジェクトの実行が開始され、受信オブジェクトがカーネルに受信処理を要求すると、カーネルは図２６の受信処理プログラムを起動する。図２６のステップ３９０では、受信オブジェクトのＯＣＢ２１０のＳＴＡＴＵＳフィールド（図１５（ｄ）の２１４）が『受信側実行待ち』または『同期完了』になっているか否かをテストする。このケースでは、この時点で送信オブジェクトからのメッセージが到着していないため、『受信側実行待ち』または『同期完了』の状態になく、「ｎｏ」となる。そこで、ステップ３９３で先に受信オブジ敵クトの要求が到着したことを示すために、ＳＴＡＴＵＳフィールド２１４を『送信側実行待ち』に設定し、カーネルにリターンする。



【００７１】「ステップＳ１３２」カーネルは、受信オブジェクトが『送信側実行待ち』になったことを知り、送信オブジェクトからメッセージが届くまで受信オブジェクトを休止させるために、図２２のＡＰＬオブジェクト終了処理を起動し、ステップ３４０でカレントケーパビリティレジスタ（ＣＣＲ）５１－２の値を図１９のタイミングｔ１３１でクリアし、ステップ３４１で受信オブジェクト再開に必要な情報をＯＣＢ２１０に退避する。

【００７２】「ステップＳ１４０」受信ＰＭ１８－２のＩＯＣ配信プログラムは、図２９に示すように、定期的にＰＭ＃２宛の受信処理ＦＩＦＯ（ＹＦ２－Ｒ（図７））を読み出すことにより受信処理要求の有無をチェックする。具体的には、図２９のステップ５００で受信処理ＦＩＦＯ（ＹＦ２－Ｒ）から要求を図１９のタイミングｔ１４０で読み出し、ステップ５０１で要求の有無をチェックする。もし、要求があれば、ステップ５０２で、引き続き同じタイミングｔ１４０で読み出した要求が指定している受信ディスクリプタリングＹＤ１２－Ｒに対して図２８の『デキュー処理』を実行し、受信ディスクリプタ（図１５（ｃ）の２００）を取り出す。ステップ５０３で、受信ディスクリプタ内のＭＢＡ（図１５（ｃ）の２０２）を参照して、図１５（ａ）で示される構造のＭＢ（Ｙ１２－１）をアクセスしてＭＢ（Ｙ１２－１）内の受信オブジェクトＩＤ（ＲＩＤ）（図１５の１８３）を取り出す。ステップ５０４で、ＲＩＤ１８３に対応する図１５（ｄ）のＯＣＢのメッセージキューに、ＭＢ（Ｙ１２－１）のアドレスＭＢＡ２０２を登録する。ステップ５０５では、この受信オブジェ説のＯＣＢ２１０のＳＴＡＴＵＳフィールド（図１５の２１４）が『送信側実行待ち』となっているか否かをテストする。ここでは、図１９のステップＳ１３１でＳＴＡＴＵＳフィールド２１４を『送信側実行待ち』と設定しているため、ステップ５０５は『ｙｅｓ』となる。これは、受信オブジェクトが送信側からメッセージを受け取る準備をして休止状態にいることを表わしているため、ステップ５０６では、受信オブジェクトのＯＣＢ２１０のＳＴＡＴＵＳフィールド２１４を『同期完了』と設定して送信要求と受信要求の両方が揃ったことを表示する。次に、ステップ５０７で、受信オブジェクトのＯＣＢ２１０をレディキューに登録して受信オブジェクトが再起動されるようにする。

【００７３】「ステップＳ１５０」受信ＰＭ１８－２のカーネルは、レディキューの中から次に実行すべきオブジェクトとして受信オブジェクトを選択し、図１９のタイミングｔ１５０で受信オブジェクトＩＤ（ＲＩＤ）を受信ＰＭ１８－２内のカレントケーパビリティレジスタ

（ＣＣＲ）５１－２に設定する。

【００７４】「ステップＳ１５１」受信オブジェクトの実行が開始され、受信オブジェクトがカーネルに受信処理を要求すると、カーネルは図２６の受信処理プログラムを起動する。図２６のステップ３９０では、受信オブジェクトのＯＣＢ２１０のＳＴＡＴＵＳフィールド２１４が『受信側実行待ち』または『同期完了』となっているか否かをテストする。このケースでは、図１９のステップＳ１４０でＳＴＡＴＵＳフィールド２１４を『同期完了』と設定しているため、ステップ３９０は『ｙｅｓ』となる。これは、メッセージが送信側から既に届いているので、直ちにメッセージを読み出してもよいことを示している。そこで、ステップ３９１では、ＯＣＢ２１０のメッセージキューにつながっているメッセージをキューから外し、そのＭＢ（Ｙ１２－１）のアドレスを取り出する。続いて、ステップ３９２で、このＭＢアドレスに対応するＭＣＲ５３－２に受信オブジェクトＩＤ（ＲＩＤ）を設定する。このタイミングは、図１９のタイミングｔ１５１で示されている。これにより、ＣＣＲ５１－２とＭＣＲ５３－２の両方に受信オブジェクトＩＤ（ＲＩＤ）が設定されたので、以後は、ＭＢ（Ｙ１２－１）が受信オブジェクト以外のＡＰＬオブジェクトからアクセスできないようにプロテクトされた。

【００７５】「ステップＳ１５２」受信オブジェクトがＭＢ（Ｙ１２－１）からタイミングｔ１５２でメッセージを読み出し、対応する処理を行う。ここで受信オブジェクトは、送信オブジェクトがメッセージを書き込んだ送信側分散共有メモリ２１－１のＭＢ（Ｘ１２－１）からでなく、そのコピーを記憶する受信側分散共有メモリ２１－２のＭＢ（Ｙ１２－１）から読み出すため、リモートメモリアクセスを行う場合に比較して極めて高速である。

【００７６】「ステップＳ１５３」受信オブジェクトは、カーネルにＭＢ解放を依頼する。カーネルは、図２４のＭＢ解放処理プログラムを起動し、図２４のステップ３６０で、ＭＢ（Ｙ１２－１）に対応するＭＣＲ５３－２を図１９のタイミングｔ１５３でクリアする。次に、図２４のステップ３６１でＭＢ（Ｙ１２－１）に対応するＭＢ管理マップエントリ（ＹＭ１２－１）を図１９のタイミングｔ１５４で『空き』に書き替える。この値の書き替えは、受信側の分散共有メモリ２１－２のみならず、同時に送信側の分散共有メモリ２１－１の同一ロケーションＸＭ１２－１に対しても行われる。その理由は、受信側分散メモリカップラ２２－２内のモジュールＩＤディレクトリ（図８の６０）に、各ＭＢ管理マップのページアドレスと、送信側ＰＭＩＤのペアが記憶されているので、受信側分散共有メモリ２１－２への書き込みが生じると、分散メモリカップラ２２－２，２２－１経由で、送信の分散共有メモリ２１－１にも書き込まれるためである。本発明では、ＭＢは常に送信ＰＭで一元管理するので、図１９のタイミングｔ１５５で、送信側のＭＢ管理マップエントリＸＭ１２－１への書き替えが完了すると、この時点以後は、いつでも送信ＰＭでＭＢ（Ｘ１２－１）を再利用することが可能である。



【００７７】「ステップＳ１５４」カーネルがＡＰＬオブジェクト終了処理を実行し、カレントケーパビリティレジスタ（ＣＣＲ）５１－２の値を図１９のタイミングｔ１５６でクリアする。以上のように、システム内ＰＭ間通信の場合には、分散メモリカップラ２２の働きにより、メッセージおよびその制御情報を送信ＰＭと受信ＰＭ間で極めて簡単に授受を行うことができるので、効率的なメッセージ通信が行える。

【００７８】（ケース３）システム間ＰＭ無中継通信

（ａ）送信システム１７－１のＰＭ１８－１内処理

送信側マルチプロセッサシステムと受信側マルチプロセッサシステムのいずれも、中継ＰＭを介さないで通信するパターンの例を説明する。図２０に示すように、マルチプロセッサシステム１７－１において送信オブジェクトの存在するプロセッサモジュール（ＰＭ）１８－１からネットワークを介して他のマルチプロセッサシステム１７－２の受信オブジェクトの存在するＰＭ１８－３にメッセージを転送する場合を考える。このケースでは、送信オブジェクトの実行終了前に、受信オブジェクトが実行開始するものとする。送信側が使用するＭＢは、図７のＸ１１－１、そのＭＢ管理マップエントリはＸＭ１１－１とする。また、この処理依頼内容の詳細は、送信ディスクリプタリングＸＤ１１－Ｓに、送信処理要求は送信ＰＭ１８－１の送信処理ＦＩＦＯ（ＸＦ１－Ｓ）に登録されるものとする。処理の流れを図２０の各ステップに沿って説明する。図２０におけるＳ１６０はＡＰＬオブジェクト起動、Ｓ１６１はＭＢ捕捉、Ｓ１６２はメッセージ書き込み、Ｓ１６３は送信処理、Ｓ１６４はＡＰＬオブジェクト終了、Ｓ１７０はＩＮＣ送信処理、Ｓ１８０はＡＰＬオブジェクト起動、Ｓ１８１は受信処理、Ｓ１８２はＡＰＬオブジェクト終了、Ｓ１９０はＩＮＣ受信処理、Ｓ２００はＡＰＬオブジェクト起動、Ｓ２０１は受信処理、Ｓ２０２はメッセージ読出し、Ｓ２０３はＭＢ解放、Ｓ２０４はＡＰＬオブジェクト終了である。

【００７９】「ステップＳ１６０」送信ＰＭ１８－１のカーネルは、ステップＳ１６０のＡＰＬオブジェクト起動の実行により、送信オブジェクトのＩＤ（ＳＩＤ）をタイミングｔ１６０でＰＭ１８－１のＣＣＲ５１－１に設定し、送信オブジェクトの実行を開始する。

【００８０】「ステップＳ１６１」送信オブジェクトはＭＢの捕捉をカーネルに要求すると、カーネルは図２３のＭＢ捕捉処理プログラムを実行する。このケースは、異なるシステムに存在する２つのＰＭ間の通信であるため、図２３のステップ３５０で検索して取り出されるシステム通信モードフィールド（図１７の２５１）の値は

41

『システム間通信』２７１となる。従って、図２３のステップ３５１では、『システム間通信』側選択され、ステップ３５５で対応するエントリの通信リンクＩＤ（図１７の２７２）と送信中継ＰＭ　ＩＤ２７３の値が取り出される。ステップ３５６で、送信中継ＰＭ　ＩＤ２７３をキーとしてＰＭルーチングテーブル２８９を検索し、ステップ３５３で送信ＰＭ１８－１と送信中継ＰＭとの通信で使用するＭＢプールとそのディスクリプタリングを決定する。このケースの送信側システムでは、ＰＭ無中継であるため送信ＰＭ１８－１と送信中継ＰＭとが一致し、ステップ３５６で、ＰＭ内送信用ＭＢプールＸ１１とそのＭＢ管理マップＸＭ１１が選択される。ステップ３５３では、ＰＭ内通信用ＭＢマップ端漆歳ＸＭ１１－１が『使用中』に書き替えられる。次に、図２３のステップ３５４で、ＭＢ（Ｘ１１－１）に対応するメモリケーパビリティレジスタ（ＭＣＲ）５３－１に、このＭＢへのアクセス権を獲得した送信オブジェクトのＩＤ（ＳＩＤ）が図２０のタイミングｔ１６２で登録される。

【００８１】「ステップＳ１６２」送信オブジェクトは、作成したメッセージを送信側分散共有メモリのＭＢ（Ｘ１１－１）にタイミングｔ１６３で書き込む。このメッセージの書き込みステップでは、ｃｃｒ５１－１とＭＣＲ５３－１の両方の値が設定されている区間は、ＭＣＲとＣＣＲにより指定されている送信オブジェクトがＭＢ（Ｘ１１－１）にアクセスすることができるが、他のＡＰＬオブジェクトはアクセスすることができない。

【００８２】「ステップＳ１６３」送信オブジェクトでは、カーネルに送信処理を依頼する。図２５の送信処理プログラムではステップ３８０でＭＢ（Ｘ１１－１）に対応するＭＣＲ５３－１をクリアする。これは、図２０のタインミングｔ１６４で示されており、これによりプロテクションウィンドウ開放期間が終了するため、以後、カーネる以外はＭＢ（Ｘ１１－１）にアクセスできなくなる。次に、図２５のステップ３８１で、通信モードをチェックする。このケースでは、『システム間ＰＭ無中継通信（システム間ＰＭ内通信）』であるため、ステップ３８７に進み、図２０のステップＳ１６１で検索一致したＰＭ　ＩＤ（図１７の２９０）、つまりＰＭ１８－１宛の送信ディスクリプタリングＸＤ１１－Ｓに送信制御情報を登録（エンキュー）し、図１４のフォーマットの処理要求を自ＰＭ１８－１宛の送信処理ＦＩＦＯ（ＸＦ１－Ｓ）（図７参照）に書き込む。これらの登録は、図２０のタイミングｔ１６５で行われる。なお、このケースでは、送信ＰＭ１８－１の送信ディスクリプタリングＸＤ１１－Ｓと送信処理ＦＩＦＯ（ＸＦ１－Ｓ）は、他の分散共有メモリ内にコピーを持たないため、分散メモリカップラによる他分散共有メモリへの書き込みは行われない。

【００８３】「ステップＳ１６４」カーネルがＡＰＬオブジェクト終了処理を起動し、カレントケーパビリティレジスタ（ＣＣＲ）５１－１の値を、図２０のタイミングｔ１６６でクリアする。

42

【００８４】「ステップＳ１７０」送信ＰＭ１８－１のＩＮＣ送信処理プログラムは、図３０に示すように、定期的に自ＰＭ宛の送信処理ＦＩＦＯ（ＸＦ１－Ｓ）を読み出すことにより要求の有無をチェックしている。具体的には、図３０のステップ５２０で、送信処理ＦＩＦＯ（ＸＦ１－Ｓ）から要求を図２０のタイミングｔ１７０で読み出し、ステップ５２１で要求の有無をチェックする。本実施例のステップＳ１７０の時点においては、既に送信処理ＦＩＦＯ（ＸＦ１－１）に要求が登録されているので、ステップ５２２で、要求が指定している送信ディスクリプタリングＸＦ１－Ｓに対して図２８の『デキュー処理』を実行する。ステップ５２３で、図１３に示す構造の送信ディスクリプタを引き続き図２０のタイミングｔ１７０で取り出し、この情報をもとにしてネットワークを介した通信に必要とする各種のプロトコル処理（例えば、ネットワークパケットのヘッダの作成、再送制御情報の作成、チェックサムの計算等）を行う。次に、ステップ５２４では、ネットワークアダプタの起動に必要なアダプタ送信ディスクリプタ（図３０のステップ５３０）を作成する。ステップ５２５では、このアダプタ送信ディスクリプタをアダプタ送信ディスクリプタリングに、図２０のタイミングｔ１７１で登録する。

【００８５】ネットワークアダプタ２４－１は、常時、アダプタ送信ディスクリプタリングを監視しており、ディスクリプタの登録があると、この情報をもとにしてネットワークが要求するサイズにメッセージを分割し、ヘッダや誤り制御情報を付加してネットワークパケットとしてネットワークに送出する。このタイミングは図２０のタイミングｔ１７２で示されている。ネットワークアダプタ２４－１は、メッセージの転送を完了するか、あるいは通信相手のシステムからメッセージ到着確認の連絡を受けると、送信側でメッセージバッファを保持しておく必要がなくなったため、図２０のタイミングｔ１７３でＭＢ（Ｘ１１－１）に対応するＭＢ管理マップのエントリＸＭ１１－１を『空き』に書き替える。その結果、この時点以降、送信ＰＭ１８－１では、いつでもこのＭＢ（Ｘ１１－１）を再利用することができる。

【００８６】（ｂ）受信システム１７－２のＰＭ１８－３内処理

「ステップＳ１８０」図２０において、受信システム１７－２のカーネルは、次に実行すべきオブジェクトとして受信オブジェクトを選択し、図２０のタイミングｔ１８０で受信オブジェクトＩＤ（ＲＩＤ）を受信ＰＭ１８－３内のカレントケーパビリティレジスタ（ＣＣＲ）５１－３に設定する。

【００８７】「ステップＳ１８１」受信オブジェクトの実行が開始され、受信オブジェクトがカーネルに受信処

理を要求すると、カーネルは図２６の受信処理プログラムを起動する。図２６のステップ３９０では、受信オブジェクトのＯＣＢ（図１５（ｄ）の２１０）のＳＴＡＴＵＳフィールド（図１４の２１４）が『受信側実行待ち』または『同期完了』になっているか否かをテストする。このケースでは、この時点で、送信オブジェクトからのメッセージは到着していないため、『受信側実行待ち』または『同期完了』の状態にはなく、『ｎｏ』となる。そこで、ステップ３９３で、先に受信オブジェクトの要求が到着したことを示すために、ＳＴＡＴＵＳフィールド２１４を『送信側実行待ち』に設定し、カーネルにリターンする。

【００８８】「ステップＳ１８２」カーネルは、受信オブジェクトが『送信側実行待ち』になったことを知り、送信オブジェクトからメッセージが届くまで受信オブジェクトを休止させるため、図２２のＡＰＬオブジェクト終了処理プログラムを起動し、ステップ３４０でカレントケーパビリティレジスタＣＣＲ５１－３の値を図２０のタイミングｔ１８１でクリアし、ステップ３４１で受信オブジェクト再開に必要な情報をＯＣＢ２１０に退避する。

【００８９】「ステップＳ１９０」図２０のタイミングｔ１７２で、送信システム１７－１からネットワークパケットが届くと、受信システム１７－２のネットワークアダプタ２４－３は、自分が管理している、分散共有メモリ２１－３内のＰＭ＃３内通信領域の空きのローカルメッセージバッファの１つに、ネットワークパケットを記憶する。メッセージが複数のネットワークパケットで分割されて送られて来る場合には、送られてきたネットワークパケットをパケットヘッダを見ながらローカルメッセージバッファ内の指定された位置に入れることによりメッセージの組み立て処理を行い、最終的に１つのメッセージに組み立てる。そして、でき上ったメッセージに関する制御情報をアダプタ受信ディスクリプタリングに登録する。ＩＮＣ受信処理プログラムは、図３１，図３２に示すように、定期的にアダプタ受信ディスクリプタリングを読み出すことにより要求の有無をチェックする。

【００９０】具体的には、図３１および図３２において、受信システム１７－２のＰＭ１８－３のＩＮＣ受信処理プログラムは、ステップ５６０でアダプタ受信ディスクリプタリングに対して、図２８の『デキュー処理』を実行し、ステップ５６１でアダプタ受信ディスクリプタリングにエントリが登録されていたか否かをチェックする。本実施例のステップＳ１９０の時点では、到着したメッセージをアダプタ受信ディスクリプタに登録済みのため、ステップ５６２でアダプタ受信ディスクリプタ（図３３の５９０）を図２０のタイミングｔ１９０で取り出し、この情報をもとにプロトコル受信処理（例えば、ネットワークパケットのヘッダのチェック、再送制



御情報の作成等）を行う。次に、ステップ５６３で、アダプタ受信ディスクリプタ中のローカルメッセージバッファアドレス（ＬＢＡ）５９１を参照してローカルメッセージバッファをアクセスし、受信オブジェクトＩＤ（ＲＩＤ）（図１５（ａ）の１８３）を取り出す。ステップ５６４では、ＲＩＤのＰＭＩＤフィールド（図１２の１７２）をキーにして、ＰＭルーチングテーブル（図１６の２８９）のＰＭ　ＩＤフィールド２９０を探索し、ＰＭ　ＩＤが一致したエントリの通信モードフィールド２９１をチェックする。

【００９１】本実施例では、自ＰＭ１８－３内の受信オブジェクト宛のメッセージであるため、通信モードフィールド２９１＝『ＰＭ内通信』となって、図３１のステップ５６５に進み、受信オブジェクトのメッセージキューにローカルメッセージバッファアドレス（ＬＢＡ）５９１を登録する。次に、ステップ５６６で、受信オブジェクトのＯＣＢ２１０のＳＴＡＴＵＳフィールド（図１５の２１４）が『送信側実行待ち』となっているか否かをテストする。このケースでは、図２０のステップＳ１８１で、ＳＴＡＴＵＳフィールド２１４を『送信側実行待ち』と設定しているため、ステップ５６６は『ｙｅｓ』となる。これは、受信オブジェクトが送信側からメッセージを受け取る準備をして休止状態にあることを表わしているので、ステップ５６７で、受信オブジェクトのＯＣＢ２１０のＳＴＡＴＵＳフィールド２１４を『同期完了』と設定して、送信要求と受信要求の両方が揃ったことを表示する。次に、ステップ５６８で受信オブジェクトのＯＣＢ２１０をレディキューに登録して、受信オブジェクトが再起動されるようにする。

【００９２】「ステップＳ２００」受信システム１７－２の受信ＰＭ１８－３のカーネルは、レディキューの中から次に実行すべきオブジェクトとして受信オブジェクトを選択し、図２０のタイミングｔ２００で受信オブジェクトＩＤ（ＲＩＤ）を受信ＰＭ１８－３内のカレントケーパビリティレジスタ（ＣＣＲ）５１－３に設定する。

【００９３】「ステップＳ２０１」受信オブジェクトの実行が開始され、受信オブジェクトがカーネルに受信処理を要求すると、カーネルは、図２６の受信処理プログラムを起動する。図２６のステップ３９０では、受信オブジェクトのＯＣＢ２１０のＳＴＡＴＵＳフィールド（図１５の２１４）が『受信側実行待ち』または『同期完了』となっているか否かをテストする。このケースでは、図２０のステップＳ１９０で、ＳＴＡＴＵＳフィールド２１４を『同期完了』と設定しているため、ステップ３９０は『ｙｅｓ』となる。これは、メッセージが送信側から既に届いているので、直ちにメッセージを読み出してもよいことを示している。そこで、ステップ３９１では、ＯＣＢ２１０のメッセージキューにつながっているメッセージを参照して、キューから外し、そのＭＢ

アドレスつまりローカルメッセージバッファアドレス（ＬＢＡ）５９１を取り出す。次に、ステップ３９２で、このＬＢＡ５９１に対応するＭＣＲ５３－３に受信オブジェクトＩＤ（ＲＩＤ）を設定する。このタイミングは、図２０のタイミングｔ２０１で示されている。これにより、ＣＣＲ５１－３とＭＣＲ５３－３の両方に受信オブジェクトＩＤ（ＲＩＤ）が設定されたので、これ以降はローカルメッセージバッファが受信オブジェクト以外のＡＰＬオブジェクトからアクセスできないようにプロテクトされた。

【００９４】「ステップＳ２０２」受信オブジェクトがローカルメッセージバッファから図２０のタイミングｔ２０２でメッセージを読み出し、対応する処理を行う。ここで、受信オブジェクトは、受信側分散共有メモリ２１－３上のローカルメッセージバッファから読み出している。

【００９５】「ステップＳ２０３」受信オブジェクトは、カーネルにローカルメッセージバッファ解放を依頼する。カーネルは、図２４のＭＢ解放処理プログラムを起動して、図２４のステップ３６０でローカルメッセージバッファに対応するＭＣＲ５３－３を図２０のタイミングｔ２０２でクリアする。次に、図２４のステップ３６１で、ローカルメッセージバッファに対応するＭＢ管理マップエントリを『空き』という値に書き替える。このタイミングは、図２０のタイミングｔ２０３に示されている。このＭＢ管理マップエントリが『空き』に書き替えられた時点以降は、ネットワークアダプタ２４－３は、このローカルメッセージバッファを再利用することができる。

【００９６】「ステップＳ２０４」カーネルがＡＰＬオブジェクト終了処理を実行して、カレントケーパビリティレジスタ（ＣＣＲ）５１－３の値を図２０のタイミングｔ２０４でクリアする。以上のように、システム間ＰＭ無中継通信の場合にも、システム内ＰＭ間通信と大差ないソフトウェアのオーバヘッドでメッセージ通信を高速に、効率よく実現することができる。

【００９７】（ケース４）システム間ＰＭ中継通信

送信マルチプロセッサシステム１７－１内の送信プロセッサモジュール（ＰＭ）１８－１内の送信オブジェクトが作成したメッセージを、宛先システムへの通信リンクを持つ送信マルチプロセッサシステム内の送信中継ＰＭ１８－２に引き継ぎ、送信中継ＰＭ１８－２がネットワークを介して他の受信マルチプロセッサシステム１７－２にメッセージを転送し、受信マルチプロセッサシステム１７－２の受信中継ＰＭ１８－３がメッセージを受信した後、受信オブジェクトの存在する受信ＰＭ１８－２にメッセージを引き継ぐ場合を仮定して、送信オブジェクトの実行終了前に、受信オブジェクトが実行開始するものとする。送信システム１７－１の送信ＰＭ１８－１が使用するＭＢは図５のＸ１２－２、そのＭＢ管理マップエントリはＸＭ１２－２、メッセージ制御情報は送信ディスクリプタリングＸＤ１２－Ｓ、送信処理要求は宛先ＰＭ１８－２用送信処理ＦＩＦＯ（ＸＦ２－Ｓ）に、それぞれ登録されるものとする。一方、受信システム１７－２の受信中継ＰＭ１８－３が使用するＭＢを、図５～図７のＺ３２－１、そのＭＢ管理マップエントリはＺＭ３２－１、メッセージ制御は受信ディスクリプタリングＺＤ３２－Ｒ、受信処理要求は宛先ＰＭ１８－１情報用受信処理ＦＩＦＯ（ＺＦ２－Ｒ）に、それぞれ登録されるものとする。処理の流れを図２１の各ステップに沿って説明する。図２１において、ステップＳ２１０はＡＰＬオブジェクト起動、ステップ２１１はＭＢ捕捉、Ｓ２１２はメッセージ書込み、Ｓ２１３は送信処理、Ｓ２２０はＩＮＣ送信、Ｓ２３０はＩＮＣ受信、Ｓ２４０はＡＰＬオブジェクト起動、Ｓ２４１は受信処理、Ｓ２４２はＡＰＬオブジェクト終了、Ｓ２５０はＩＯＣ配信、Ｓ２６０はＡＰＬオブジェクト起動、Ｓ２６１は受信処理、Ｓ２６２はメッセージ読出し、Ｓ２６３はＭＢ解放、Ｓ２６４はＡＰＬオブジェクト終了である。



【００９８】（ａ）送信システム１７－１の送信ＰＭ１８－１の処理

「ステップＳ２１０」送信マルチプロセッサシステム１７－１の送信ＰＭ１８－１のカーネルは、図２１のステップＳ２１０のＡＰＬオブジェクト起動プログラムを実行し、送信オブジェクトのＩＤ（ＳＩＤ）をタイミングＳ２１０でＰＭ１８－１のＣＣＲ５１－１に設定し、送信オブジェクトの実行を開始する。

【００９９】「ステップＳ２１１」送信オブジェクトはＭＢの捕捉をカーネルに要求すると、カーネルは図２３のＭＢ捕捉処理プログラムを実行する。このケースでは、送信システム内送信ＰＭ１８－１から送信システム内中継ＰＭ１８－２への通信であるため、図２３のステップ３５０で検索して取り出されるシステム通信モードフィールド（図１６の２５１）の値は『システム間通信』２７１となる。従って、図２３のステップ３５１では、『システム間通信』側が選択され、ステップ３５５で対応するエントリの通信リンクＩＤ（図１７の２７２）と送信中継ＰＭ　ＩＤ２７３の値が取り出される。このケースでは、送信中継ＰＭ　ＩＤ２７３＝『ＰＭ１８－２』であるため、ステップ３５５，３５６では、『ＰＭ１８－２』の値をキーとしてＰＭルーチングテーブル２８９の検索が行われる。その結果、ＰＭ　ＩＤフィールド２９０＝『ＰＭ１８－２』となっているエントリが一致し、このエントリに対応するＭＢ管理マップアドレス＝『ＰＭ１８－１／ＰＭ１８－２間通信ＭＢ（Ｘ１２）管理マップベースアドレス３１２』が、ＰＭルーチングテーブル２８９から取り出される。

【０１００】ステップ３５３で、ＰＭ内通信用ＭＢ（Ｘ１２）のうちの空きＭＢ（Ｘ１２－２）が選択され、そのＭＢ管理マップエントリＸＭ１２－２を『使用中』に

書き替える。その書き替えは、図２１のタイミングｔ２１１で、送信側分散共有メモリ２１－１上のＭＢ管理マップエントリ（ＸＭ１２－２）およびそれと同一アドレスを持つ、送信中継側分散共有メモリ２１－２のＭＢ管理マップエントリ（ＹＭ１２－２）の両方に対して行われる。その具体的方法は、図１９のステップＳ１２１で説明したので、ここでは省略する。次に、図２３のステップ３５４で、ＭＢ（Ｘ１２－２）に対応するメモリケーパビリティレジスタ（ＭＣＲ）５３－１に、このＭＢへのアクセス権を獲得した送信オブジェクトのＩＤ（ＳＩＤ）を図２１のタイミングｔ２１２で登録する。

【０１０１】「ステップＳ２１２」送信オブジェクトは、作成したメッセージを送信側分散共有メモリのＭＢ（Ｘ１２－２）にタイミングｔ２１３で書き込むと、この書き込みデータが分散メモリカップラ２２－１により送信中継側分散共有メモリ２１－２の同一アドレスのＭＢ（Ｙ１２－２）にも伝搬して書き込まれる。その操作は、図１９のｔ１２１で説明したので、省略する。なお、このメッセージ書き込みステップでは、ＣＣＲ５１－１とＭＣＲ５３－１の両方の値が設定されている期間中、ＭＣＲとＣＣＲが指定している送信オブジェクトはＭＢ（Ｘ１２－２）にアクセスすることができるが、他のＡＰＬオブジェクトはアクセスすることができない。また、送信中継ＰＭ１８－２にも、各ＭＢに対してＭＣＲ５３－２が用意されているが、ＭＣＲ５３－２には値が設定されていないため、もし、送信中継ＰＭ１８－２内のアプリケーションオブジェクトからＭＢ（Ｙ１２－２）へのアクセスがあった場合にも、ＭＣＲ５３－２とＣＣＲ５１－２の不一致が生じて送信中継ＰＭ１８－２内での不正アクセスを検出することができる。

【０１０２】「ステップＳ２１３」送信オブジェクトは、カーネルに送信処理を依頼する。送信処理プログラムは、図２５に示すように、先ずステップ３８０でＭＢ（Ｘ１２－２）に対応するＭＣＲ５３－１を図２１のタイミングｔ２１４でクリアする。これにより、プロテクションウィンドウ開放期間が終了するので、以降はカーネル以外は、ＭＢ（Ｘ１２－２）にアクセスできなくなる。次に、図２５のステップ３８１で、通信モードが『システム間ＰＭ中継送信』であることを判定し、ステップ３８７では、図２１のステップＳ２１１で検索が一致したＰＭ　ＩＤ（図１７の２９０）、つまりＰＭ１８－２宛の送信ディスクリプタリングＸＤ１２－Ｓに、送信制御情報を登録（エンキュー）し、図１４のフォーマットの処理要求をＰＭ１８－２宛の送信処理ＦＩＦＯ（ＸＦ２－Ｓ）（図７参照）に書き込む。この書き込み動作は、図２１のタイミングｔ２１５で示されている。その結果、分散メモリカップラ２２－１により送信ディスクリプタリング（ＸＤ１２－Ｓ）への書き込みデータが送信中継側の分散共有メモリ２１－１上の同一アドレスロケーションＹＤ－Ｓにコピーされる。同じく、送信



処理ＦＩＦＯ（ＸＦ２－Ｓ）の書き込みデータが、送信中継ＰＭ宛送信ＦＩＦＯにもコピーされる。これにより、メッセージ送信に必要な全ての情報が受信中継ＰＭ１８－２に伝達される。

【０１０３】「ステップＳ２１４」カーネルがＡＰＬオブジェクト終了処理を起動し、カレントケーパビリティレジスタ（ＣＣＲ）５１－１の値を図２１のタイミングｔ２１６でクリアする。

【０１０４】（ｂ）送信システム１７－１の送信中継ＰＭ１８－２側の処理

「ステップＳ２２０」送信中継ＰＭ１８－２のＩＮＣ送信処理プログラムは、図３０に示すように、定期的に自ＰＭ宛の送信処理ＦＩＦＯ（ＹＦ２－Ｓ）を読み出すことにより要求の有無をチェックしている。具体的には、図３０のステップ５２０で、送信処理ＦＩＦＯ（ＹＦ２－Ｓ）から要求を図２１のタイミングｔ２２０で読み出し、ステップ５２１で要求の有無をチェックする。もし、要求があれば、ステップ５２２で読み出した要求が指定している送信ディスクリプタリングに対して図２８の『デキュー処理』を実行する。本実施例では、ステップＳ２１３で既に送信処理ＦＩＦＯ（ＸＦ２－Ｓ）に要求が登録されているので、ステップ５２３で、図１５（ｃ）に示す構造の送信ディスクリプタエントリを図２１のタイミングｔ２２０で取り出し、この情報をもとにネットワークを介して通信に必要な各種のプロトコル処理（例えば、ネットワークパケットのヘッダの作成、再送制御情報の作成、チェックサムの計算等）を行う。次に、ステップ５２４では、ネットワークアダプタの起動に必要なアダプタ用送信ディスクリプタを作成する。ステップ５２５では、このアダプタ送信ディスクリプタをアダプタ送信ディスクリプタリングに図２０のタイミングｔ２２１で登録する。

【０１０５】ネットワークアダプタ２４－２は、常時、アダプタ用送信ディスクリプタリングを監視しており、ディスクリプタの登録があると、この情報をもとにしてネットワークが要求するサイズにメッセージを分割し、ヘッダや誤り制御情報を付加してネットワークパケットとしてネットワークに送出する。このタイミングは、図２１のタイミングｔ２２２で示されている。ネットワークアダプタ２４－２はメッージの転送を完了するか、または通信相手のシステムからメッセージ到着確認の連絡を受けると、送信側でメッセージバッファを保持しておく必要がなくなったので、図２１のタイミングｔ２２３でＭＢ管理マップエントリＹＭ１２－２を『空き』に書き替えると、送信側分散共有メモリにも伝搬してＭＢ管理マップエントリＹＭ１２－２がタイミングｔ２２４で書き替えられる。その結果、これ以降は、送信ＰＭ１８－１は、対応するＭＢ（Ｘ１２－２）を再利用することができる。なお、送信中継ＰＭ１８－２では、送信ＰＭ１８－１のように、ＭＣＲ５３、ＣＣＲ５１にケーパビ

リティを設定していないが、もし送信中継ＰＭ１８－２内のＡＰＬオブジェクトからＭＢ（Ｙ１２－２）に不正アクセスが行われた場合、ＭＢ（Ｙ１２－２）対応のＭＣＲ５３－２に値が未設定であることにより、不正アクセスが検出できる。

【０１０６】（ｃ）受信システム１７－２の受信中継ＰＭ１８－３側の処理

「ステップＳ２３０」送信システム１７－１からネットワークパケットが届くと、受信システム１７－２の受信中継ＰＭ１８－３のネットワークアダプタ２４－３は、予め自分が管理している、分散共有メモリ２１－３上のＰＭ＃３内通信領域内の空きのローカルメッセージバッファの１つに、ネットワークパケットを記憶する。メッセージが複数のネットワークパケットで分割されて送られて来る場合には、ネットワークアダプタ２４－３は、受信したパケットをパケットヘッダを見ながらローカルメッセージバッファ内の指定された位置に入れることによりメッセージの組み立て処理を行って、最終的に１つのメッセージに組み立てる。そして、完成したメッセージに関する制御情報をアダプタ受信ディスクリプタリングに登録する。受信中継ＰＭ１８－３のＩＮＣ受信処理プログラムは、図３１，図３２に示すように、定期的にアダプタ受信ディスクリプタリングを読み出すことにより要求の有無をチェックする。

【０１０７】具体的には、図３１において、ＩＮＣ受信処理プログラムのステップ５６０で、アダプタ受信ディスクリプタリングに対して図２８の『デキュー処理』を実行し、ステップ５６１でアダプタ受信ディスクリプタリングにエントリが登録されていたか否かをチェックする。図２１のステップ２３０の時点においては、到着したメッセージをアダプタ受信ディスクリプタに登録済みであるため、ステップ５６２では、アダプタ受信ディスクリプタエントリ（図３３の５９０）を図２１のタイミングｔ２３０で取り出し、この情報をもとにプロトコル受信処理（例えば、ネットワークパケットのヘッダのチェック、再送制御情報の作成等）を行う。次に、ステップ５６３では、アダプタ受信ディスクリプタ中のローカルメッセージバッファアドレスＬＢＡ５９１を参照することによりローカルメッセージバッファをアクセスし、受信オブジェクトＩＤ（ＲＩＤ）を取り出す。ステップ５６４では、ＲＩＤのＰＭ　ＩＤフィールド（図１２の１７２）をキーとして、ＰＭルーチングテーブル２８９のＰＭ　ＩＤフィールド＝『受信システム１７－２の受信ＰＭ１８－２』となっているエントリと一致するので、通信モードフィールド２９１＝『ＰＭ間通信』となり、ステップ５７０に進む。

【０１０８】図３２のステップ５７０で、一致したエントリに対応するＭＢ管理マップアドレス＝『ＰＭ１８－３／ＰＭ１８－２間通信ＭＢ（Ｘ３２）管理マップＺＭ３２のベースアドレス』をＰＭルーチングテーブル２８９から取り出す。ステップ５７１では、ＭＢ管理マップＺＭ３２を参照して空きＭＢ（Ｚ３２－１）を選択し、そのＭＢ管理マップエントリＺＭ３２－１を図２１のタイミングｔ２３１で『使用中』に書き替える。ＺＭ３２－１を書き替えると、受信システムの分散メモリカップラ２２－３、２２－２経由で、ＺＭ３２－１と同一アドレスを持つ受信側分共有メモリ２１－２のＭＢ管理マップエントリＹＭ３２－１も『使用中』に書き替えられる。次に、ステップ５７２で、ローカルメッセージバッファにあるメッセージをＭＢ（Ｚ３２－１）に図２１のタイミングｔ２３２でコピーする。このコピー操作を実行すると、分散メモリカップラ２２－３，２２－２経由でメッセージが受信ＰＭの分散共有メモリ２１－２のＭＢ（Ｙ３２－１）にも書き込まれる。つまり、このコピー操作は、受信側分散共有メモリ２１－２に書き込みデータを送るために行ったものである。

【０１０９】図３２のステップ５７３，５７４，５７５では、送信中継ＰＭ１８－３から受信専１８－２宛の受信ディスクリプタリングＺＤ３２－Ｒにメッセージ制御情報を図２１のタイミングｔ２３３で書き込むとともに、受信処理要求を受信ＰＭ１８－２宛の受信処理ＦＩＦＯ（ＺＦ２－Ｒ）に書き込む。その結果、分散メモリカップラ２２－３，２２－１により転送されるので、受信ディスクリプタリングＺＤ３２－Ｒの内容が受信ＰＭ１８－２の分散共有メモリ２１－２の同一アドレスロケーションであるＹＤ３２－Ｒに、また受信処理ＦＩＦＯ（ＺＦ２－Ｒ）の内容が受信ＰＭ１８－２の分散共有メモリ２１－２の同一アドレスロケーションであるＹＦ２－Ｒに、それぞれコピーされる。これにより、メッセージ送信に必要な情報が受信ＰＭ１８－３に伝達された。なお、受信中継ＰＭ１８－３においては、ＭＣＲ５３、ＣＣＲ５１にケーパビリティを設定していないが、もし受信中継ＰＭ１８－３内のＡＰＬオブジェクトからＭＢ（Ｚ２３－１）に不正アクセスが行われた場合には、ＭＢ（Ｚ３２－１）対応するＭＣＲ５３－３に値が未設定であることにより、不正アクセスは検出できる。

【０１１０】（ｄ）受信システム１７－２の受信ＰＭ１８－２側の処理

「ステップＳ２４０」図２１のステップＳ２４０で、カーネルは次に実行すべきオブジェクトとして受信オブジェクトを選択し、図２１のタイミングｔ２４０で受信オブジェクトＩＤ（ＲＩＤ）を受信ＰＭ１８－２内のカレントケーパビリティレジスタＣＣＲ５１－２に設定する。

【０１１１】「ステップＳ２４１」受信オブジェクトがカーネルに受信処理を要求すると、カーネルは図２６の受信処理プログラムを起動する。図２６のステップ３９０では、受信オブジェクトのＯＣＢ（図１５（ｄ）の２１４）が『受信側実行待ち』または『同期完了』になっているか否かをテストする。このケースでは、この時点

51

で送信オブジェクトからのメッセージが到着していないので、『受信側実行待ち』あるいは『同期完了』の状態になく、『ｎｏ』となる。そこで、ステップ３９３で、先に受信オブジェクトの要求が到着したことを示すために、ＳＴＡＴＵＳフィールド２１４を『送信側実行待ち』に設定し、カーネルにリターンする。

【０１１２】「ステップＳ２４２」カーネルは、受信オブジェクトが『送信側実行待ち』になったことを知り、送信オブジェクトからメッセージが届くまで受信オブジェクトを休止させるため、カーネルが図２２のＡＰＬオブジェクト終了処理を起動し、ステップ３４０でカレントケーパビリティレジスタ（ＣＣＲ）５１－２の値を図２１のタイミングｔ２４１でクリアし、ステップ３４１で受信オブジェクト再開に必要な情報をＯＣＢ２１０に退避する。

【０１１３】「ステップＳ２５０」受信ＰＭ１８－２のＩＯＣ配信プログラムは、図２９に示すように、定期的にＰＭ＃２宛の受信処理ＦＩＦＯ（ＹＦ２－Ｒ）を読み出すことにより受信処理要求の有無をチェックする。具体的には、図２９のステップ５００で、ＦＩＦＯから要求を図２１のタイミングｔ２５０で読み出し、ステップ５０１で要求の有無をチェックする。もし、要求があれば、ステップ５０２で、読み出した要求が指定している受信ディスクリプタリング（ＹＤ１２－Ｒ）に対して図２８の『デキュー処理』を実行する。ステップ５０３では、図１５（ｃ）の受信ディスクリプタエントリを同じタイミングｔ２５０で取り出し、そのエントリのＭＢＡ（図１５（ｃ）の２０２）を参照して、図１５（ａ）で示される構造のＭＢ（Ｙ３２－１）をアクセスする。そして、ＭＢ（Ｙ３２－１）内の受信オブジェクトＩＤ（ＲＩＤ）（図１５（ａ）の１８３）を取り出す。ステップ５０４では、ＲＩＤ１８３に対応するＯＣＢ（図１５の２１０）のメッセージキューにＭＢ（Ｙ３２－１）のアドレスＭＢＡ２０２を登録する。

【０１１４】図２９のステップ５０５では、この受信オブジェクトのＯＣＢ２１０のＳＴＡＴＵＳフィールド２１４が『送信側実行待ち』となっているか否かをテストする。このケースでは、図２１のステップＳ２４１で、ＳＴＡＴＵＳフィールド２１４を『送信側実行待ち』と設定しているので、ステップ５０５は『ｙｅｓ』となる。これは、受信オブジェクトが送信側からメッセージを受け取る準備をして休止状態にいることを表わしている。従って、ステップ５０６では、受信オブジェクトのＯＣＢ２１０のＳＴＡＴＵＳフィールド２１４を『同期完了』と設定して、送信要求と受信要求の両方が揃ったことを表示する。次に、ステップ５０７では、受信オブジェクトのＯＣＢ２１０をレディキューに登録して、受信オブジェクトが再起動されるようにする。

【０１１５】「ステップＳ２６０」カーネルは、レディキューの中から次に実行すべきオブジェクトとして受信

52

オブジェクトを選択し、図２１のタイミングｔ２６０で、受信オブジェクトＩＤ（ＲＩＤ）を受信ＰＭ１８－２内のカレントケーパビリティレジスタ（ＣＣＲ）５１－２に設定する。

【０１１６】「ステップＳ２６１」受信オブジェクトの実行が開始され、受信オブジェクトがカーネルに受信処理を要求すると、カーネルは図２６の受信処理プログラムを起動する。図２６のステップ３９０では、受信オブジェクトのＯＣＢ２１０のＳＴＡＴＵＳフィールド２１４が『受信側実行待ち』または『同期完了』となっているか否かをテストする。このケースでは、図２１のステップＳ２５０でＳＴＡＴＵＳフィールド２１４を『同期完了』と設定しているので、ステップ３９０は『ｙｅｓ』となる。これは、メッセージが送信側から既に届いているので、直ちにメッセージを読み出してもよいことを示している。そこで、ステップ３９１で、ＯＣＢ２１０のメッセージキューにつながっているメッセージを参照して、キューから外し、そのＭＢ（Ｙ３２－１）のアドレスを取り出す。次に、ステップ３９２では、このＭＢアドレスに対応するＭＣＲ５３－２に受信オブジェクトＩＤ（ＲＩＤ）を設定する。このタイミングは、図２１のタイミングｔ２６１で示されている。これにより、ＣＣＲ５１－２とＭＣＲ５３－２の両方に受信オブジェクトＩＤ（ＲＩＤ）が設定されたので、それ以降は、ＭＢ（Ｙ３２－１）が受信オブジェクト以外のＡＰＬオブジェクトからアクセスできないようにプロテクトされた。

【０１１７】「ステップＳ２６２」受信オブジェクトがＭＢ（Ｙ３２－１）から、図２１のタイミングｔ２６２で受信側分散共有メモリ２１－２のＭＢ（Ｙ３２－１）からメッセージを読み出し、対応する処理を行う。ＭＢ（Ｙ３２－１）からの読み出しは、自プロセッサモジュール内に閉じたローカルなメモリアクセスであるため、リモートメモリアクセスを行う場合に比較して極めて高速である。

【０１１８】「ステップＳ２６３」受信オブジェクトは、カーネルにＭＢ解放を依頼する。カーネルは、図２４のＭＢ解放処理プログラムを起動し、図２４のステップ３６０で、ＭＢ（Ｙ３２－１）に対応するＭＣＲ５３－２をクリアする。このタイミングは、図２１のタイミングｔ２６３に示されている。次に、図２４のステップ３６１では、ＭＢ（３２－１）に対応するＭＢ管理マップエントリＹＭ３２－１を『空き』という値に書き替える。このタイミングは、図２１のタイミングｔ２６４に示されている。この値の書き替えは、受信側の分散共有メモリ２１－２のみならず、同時に分散メモリカップラ２２－２，２２－１経由で、受信中継側分散共有メモリ２１－３の同一番地ＺＭ３２－１に対しても行われる。図２１のタイミングｔ２６５で、受信中継ＰＭ１８－３のＭＢ管理マップエントリＺＭ３２－１が『空き』に書

き替えられた時点以降は、受信中継ＰＭ１８－３のＭＢ（Ｚ３２－１）をいつでも再利用することができる。

【０１１９】「ステップＳ２６４」カーネルがＡＰＬオブジェクト終了処理を実行して、カレントケーパビリティレジスタ（ＣＣＲ）５１－２の値を図２１のタイミングｔ２６６でクリアする。以上のように、システム間ＰＭ中継通信においても、分散メモリカップラ２２の働きにより、送受信中継ＰＭを介してもメッセージおよびその制御情報を送信ＰＭと受信ＰＭ間で極めて簡単、かつ高速に転送することができるので、効率的なメッセージ通信が実現される。本実施例においては、図２９のＩＯＣ配信処理、図３０のＩＮＣ送信処理、図３１，図３２のＩＮＣ受信処理は、分散共有メモリの特定番地を読み出すことにより要求を検出する、いわゆるポーリング型の処理方式で説明したが、これらを図８に示す割り込みデータタイプのパケットと図１の割り込み制御部４２の機能を用いて割り込み型の処理方式で実現することも可能である。

【０１２０】（本発明のまとめ）このように、本発明においては、各プロセッサモジュールが分散共有メモリと分散メモリカップラとを持ち、分散メモリカップラは、その分散共有メモリのアドレスを共有しているプロセッサモジュール情報を記憶する。そして、送信側のプロセッサモジュールの分散共有メモリの共有アドレスロケーションへの書き込みが発生した時点で、分散メモリカップラが、アドレスを共有する他のプロセッサモジュールに書き込み情報を転送し、書き込み情報を受けた受信側の分散メモリカップラが受信側の分散共有メモリへコピーを行う。この転送動作は、ソフトウェアの介在なしで実行されるので、従来では、分散共有メモリ間転送用としてわざわざ必要であったＤＭＡ機構制御用プログラムの実行は不要となり、ソフトウェアオーバヘッドを大幅に削減することができる。また、ＤＭＡ機構を使用する従来の方式では、転送データが全部揃ってから転送を開始するのに対して、本発明では、送信元の分散共有メモリへの書き込みが発生した時点で、相手側の分散共有メモリへの転送が開始されるので、メッセージが送信側から受信側に伝達されるまでの遅延時間（レイテンシー）も大幅に削減される。さらに、分散メモリカップラに登録するプロセッサモジュール情報に応じて、一方向性通信、両方向通信、放送型通信等の各種の通信パターンを自由に実現することができるので、柔軟性が高い。

【０１２１】また、本発明における送信プロセッサモジュールと受信プロセッサモジュール間のメッセージ通信では、送信プロセッサモジュールでメッセージバッファ、メッセージの制御情報であるＭＢディスクリプタ、メッセージバッファ使用状態を示すＭＢ管理マップを分散共有メモリに配置している。送信プロセッサモジュールが、これらの情報を送信側分散共有メモリに書き込み、受信側プロセッサモジュールが分散共有メモリの同



じアドレスロケーションから読み出すだけでよいので、プロセッサモジュール間の通信の同期を簡単かつ容易に行うことができる。また、受信側プロセッサモジュールが受信側分散共有メモリのＭＢディスクリプタとＭＢ管理マップに返事を書き込み、送信側プロセッサモジュールが受信側分散共有メモリの同じアドレス位置のＭＢディスクリプタとＭＢ管理マップから読み出すことにより、ＭＢディスクリプタやＭＢに関する状態を簡単に送信側に伝達することができ、ＭＢディスクリプタやＭＢ再使用すればよいか等を簡単に知ることができる。

【０１２２】さらに、本発明では、受信側プロセッサモジュールの分散共有メモリの、あるアドレスロケーションをＦＩＦＯ構造にしておくことにより、送信側プロセッサモジュールが自分の分散共有メモリに書き込んだ情報が、受信側分散共有メモリのＦＩＦＯに順次書き込まれる。その結果、複数の送信側プロセッサモジュールが同時に、それぞれの分散共有メモリの同一番地に書き込みを行った場合にも、受信側の分散共有メモリのＦＩＦＯで競合整理が行われてＦＩＦＯに順次書き込まれ、ソフトウェアによる競合処理を全く必要としない。受信側は、分散共有メモリ内の要求登録ＦＩＦＯを順次読み出すのみで、実際に発生した要求のみを効率よく取り出すことができ、従来のような多数の監視点を順次サーチしていくための処理オーバヘッドを大幅に削減することができる。このように、例えば同一メモリエリアへの同時アクセス等、従来、マルチプロセッサシステムで大きな問題となっていたプロセッサ間のリソース競合を回避することができるので、多数のプロセッサを組み合わせた超並列コンピュータシステムを構築した場合でも、プロセッサの台数に応じた大きな処理能力を持たせることが可能である。

【０１２３】

【発明の効果】以上説明したように、本発明によれば、カーネル等のシステムソフトウェアのオーバヘッドを削減することができるとともに、転送遅延時間が短く、処理効率の高いメッセージ転送を実現することが可能である。

【図面の簡単な説明】

【図１】本発明の一実施例を示すプロセッサモジュールの内部構成図である。

【図２】従来のマルチプロセッサシステムの構成図である。

【図３】従来のプロセッサ間メッセージ通信方法を示すシーケンスチャートである。

【図４】本発明の一実施例を示すマルチプロセッサシステムの構成図である。

【図５】本発明における分散共有メモリのデータ配置を示す図である。

【図６】同じく、分散共有メモリのデータ配置の残りを示す図である。

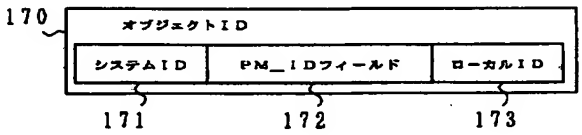
55

【図７】同じく、分散共有メモリのデータ配置の残りを示す図である。
【図８】本発明におけるモジュールＩＤ管理部の構成図である。
【図９】本発明におけるパケット送信レジスタと、パケット受信レジスタのデータフォーマット図である。
【図１０】本発明におけるシステム内メッセージ通信の概要を示す図である。
【図１１】本発明におけるシステム間メッセージ通信の概要を示す図である。
【図１２】本発明におけるオブジェクトＩＤのフォーマット図である。
【図１３】本発明における送信ディスクリプタと送信処理要求のメモリ配置図である。
【図１４】本発明における送信処理要求のフォーマット図である。
【図１５】本発明における各種制御データの構造を示す図である。
【図１６】本発明におけるレディキューの構造を示す図である。
【図１７】本発明におけるシステムルーチングテーブルとＰＭルーチングテーブルの構造図である。
【図１８】本発明におけるシステム内ＰＭ内通信のタイムチャートである。
【図１９】本発明におけるシステム内ＰＭ間通信のタイムチャートである。
【図２０】本発明におけるシステム間ＰＭ無中継通信のタイムチャートである。
【図２１】本発明にかけるシステム間ＰＭ中継通信のタイムチャートである。
【図２２】本発明におけるアプリケーションオブジェクトの起動、終了処理のフローチャートである。
【図２３】本発明におけるＭＢ捕捉処理のフローチャートである。
【図２４】本発明におけるＭＢ解放処理のフローチャートである。
【図２５】本発明における送信処理のフローチャートである。
【図２６】本発明における受信処理のフローチャートである。
【図２７】本発明におけるエンキュー処理のフローチャートである。

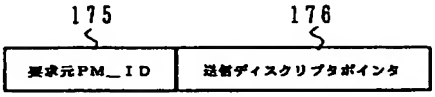
56

【図２８】本発明におけるデキュー処理のフローチャートである。
【図２９】本発明におけるＩＯＣ配信処理のフローチャートである。
【図３０】本発明におけるＩＮＣ送信処理のフローチャートおよびアダプタ送信ディスクリプタの構造図である。
【図３１】本発明におけるＩＮＣ受信処理のフローチャートである。
【図３２】同じく、ＩＮＣ受信処理の残りのフローチャートである。
【図３３】本発明におけるアダプタ受信ディスクリプタの構造図である。
【符号の説明】
１７・・マルチプロセッサシステム、
１８－１，１８－２，１８－３・・プロセッサモジュール、
１９－１，１９－２，１９－３・・プロセッサ、
２０－１，２０－２，２０－３・・ローカルメモリ、
２１－１，２１－２，２１－３・・分散共有メモリ、
２２－１，２２－２，２２－３・・分散メモリカップラ、
２３－１，２３－２，２３－３・・分散メモリプロテクタ、
２４－１，２４－２，２４－３・・ネットワークアダプタ、
２５・・プロセッサモジュール間通信路、
２６・・通信ネットワーク、３４・・モジュールＩＤ管理部、
３５・・パケット送信レジスタ、３６・・パケット送信バッファ、
３７・・パケット受信バッファ、３８・・パケット受信レジスタ、
５０・・ＭＣＲメモリ（メモリケーパビリティレジスタメモリ）、
５１・・ＣＣＲ（カレントケーパビリティレジスタ）、
５３・・ＭＣＲ、６０・・モジュールＩＤディレクトリ、
６１・・ＣＡＭ (Content Addressable Memory)、
６２・・データメモリ部、

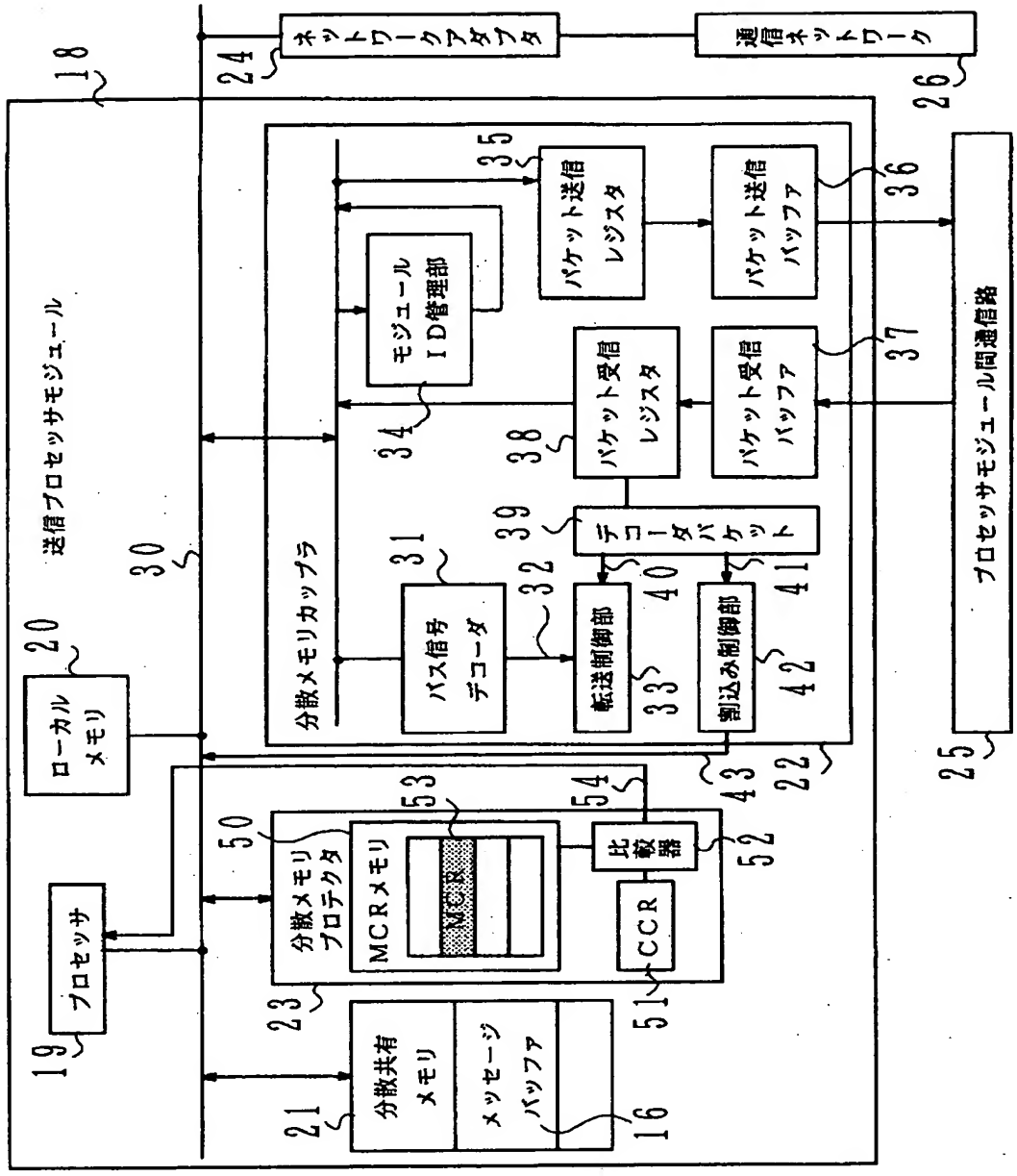
【図１２】



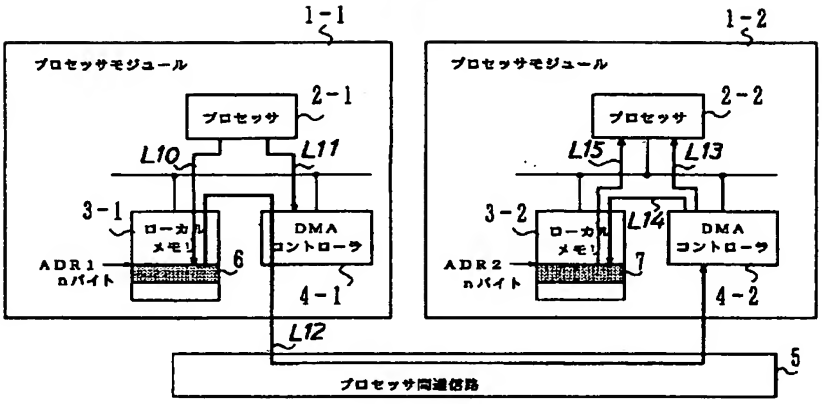

【図１４】



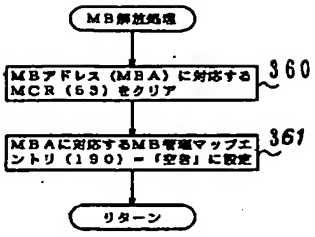
【図１】


送信プロセッサモジュール
18
ネットワークアダプタ
24
通信ネットワーク
26
プロセッサモジュール間通信路
25
ローカルメモリ
20
プロセッサ
19
分散メモリカップラ
30
モジュールID管理部
34
パケット送信レジスタ
35
パケット送信バッファ
36
パケット受信レジスタ
38
パケット受信バッファ
37
デコーダパケット
39
パス信号デコーダ
31
32
転送制御部
33
40
割込み制御部
41
42
43
22
分散メモリプロテクタ
50
MCRメモリ
53
MCR
比較器
54
52
CCR
51
23
分散共有メモリ
21
メッセージバッファ
16

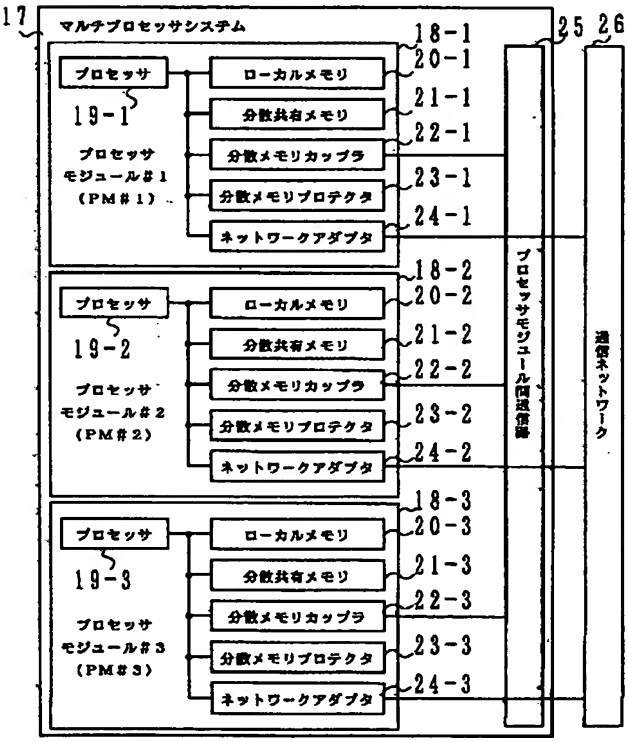
【図２】



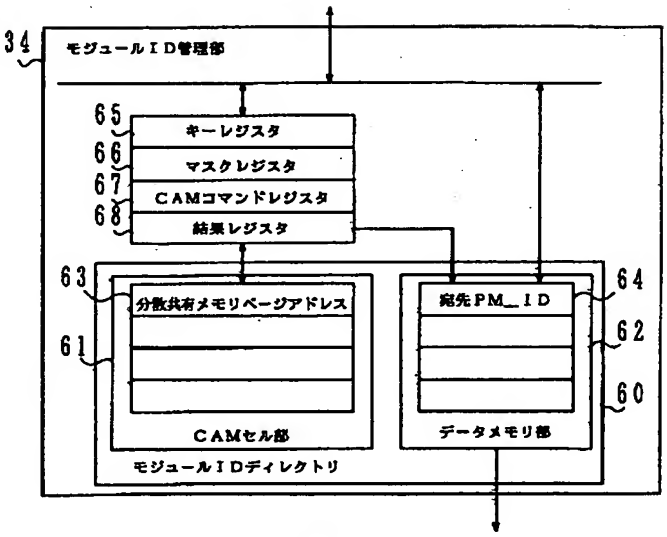

【図２４】



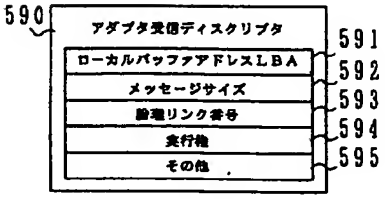

【図４】



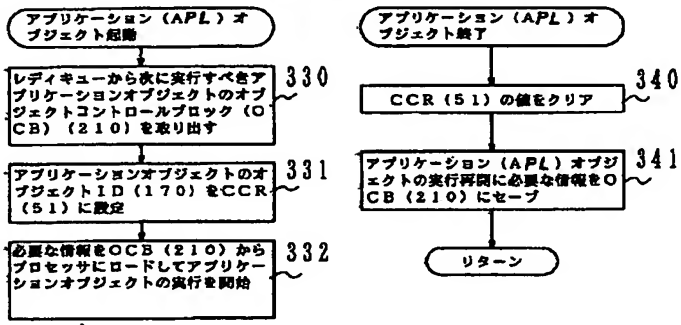

【図８】



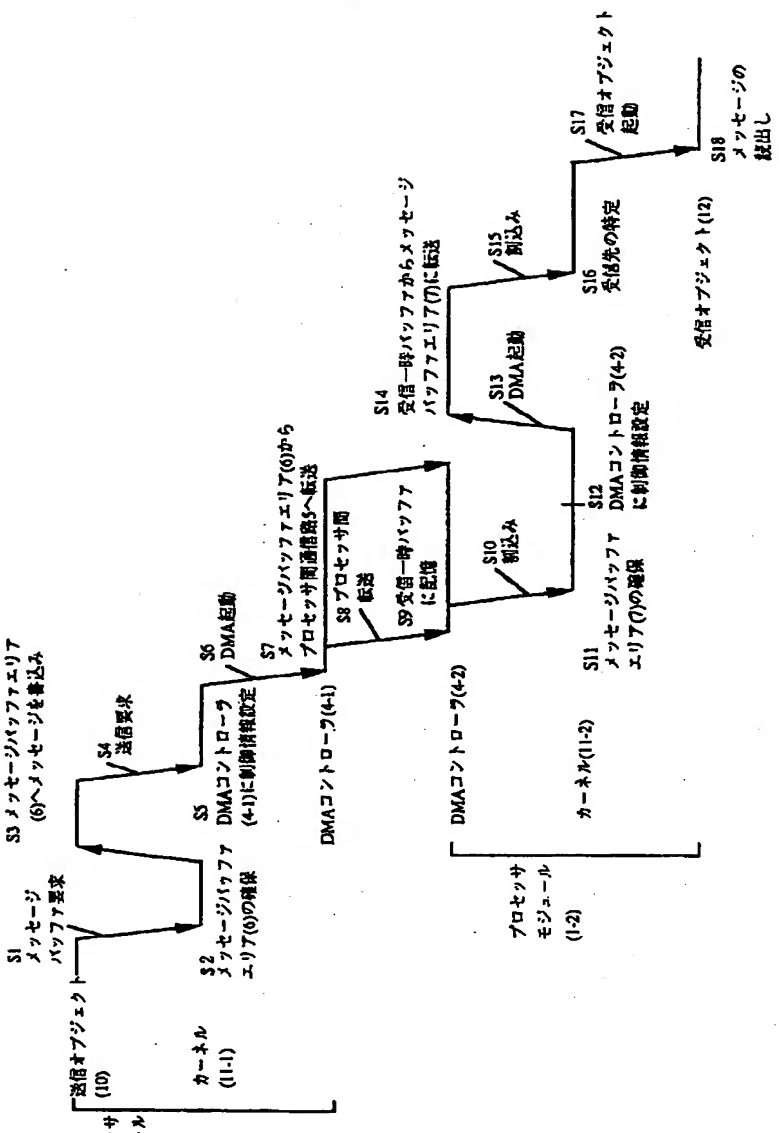

【図３３】



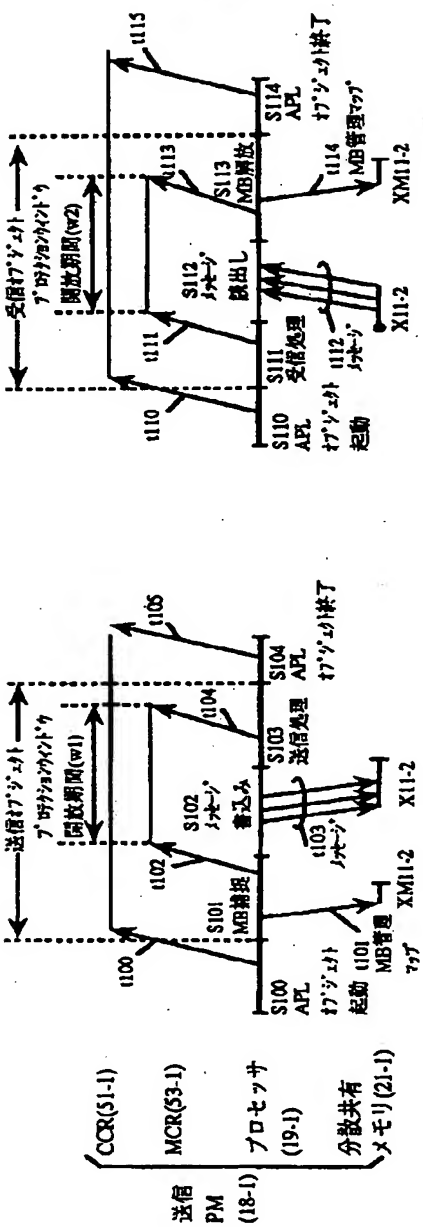

【図２２】



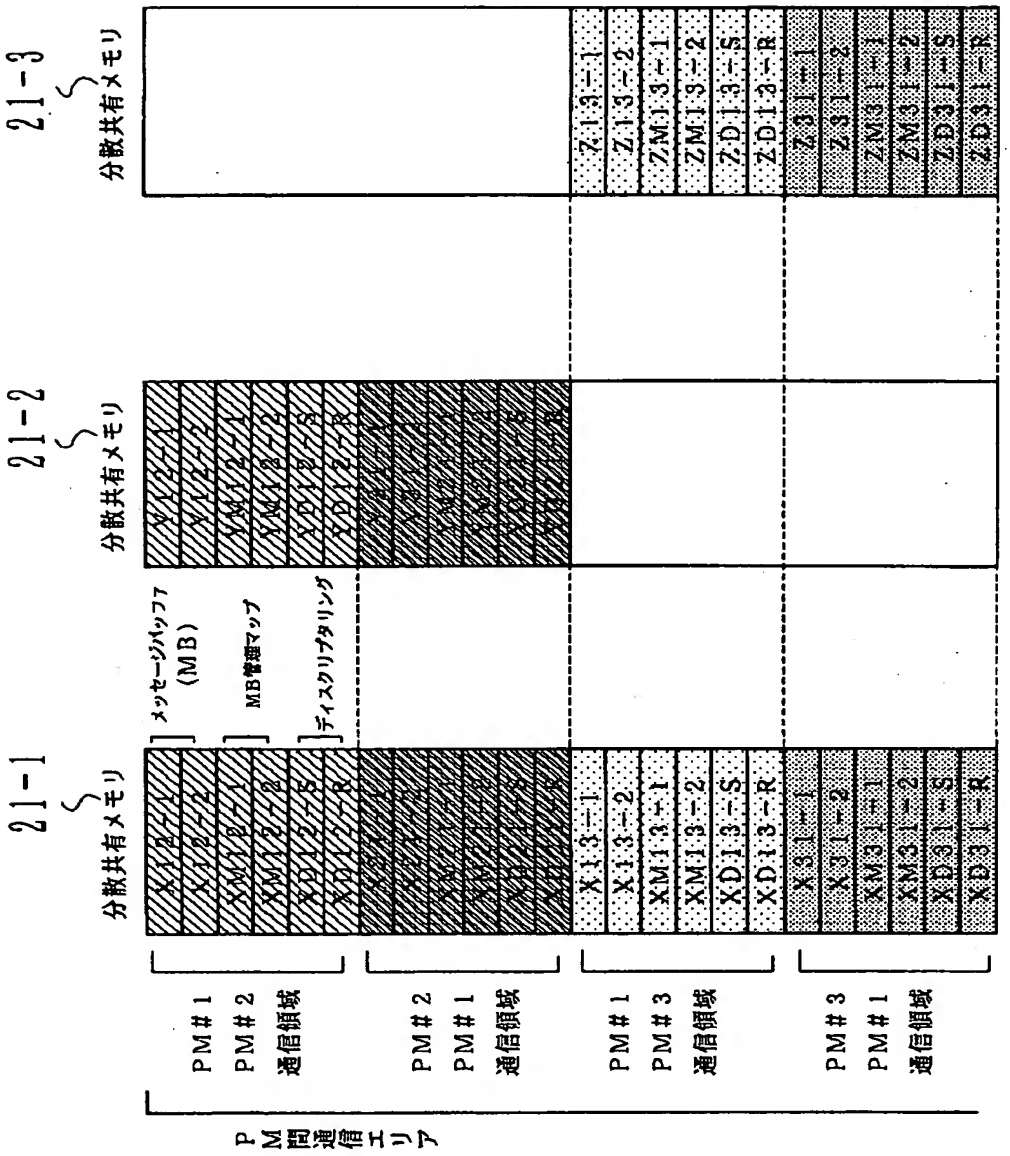
【図３】


S1 メッセージバッファエリア
(6)へメッセージを書込み
S3 メッセージバッファエリア
バッファ要求
S4 送信要求
S5 DMAコントローラ
(4-1)に制御情報設定
S6 DMA起動
S7 メッセージバッファエリア(6)から
プロセッサ間通信路5へ転送
S8 プロセッサ間
転送
S9 受信一時バッファ
に記憶
S10 割込み
S11 メッセージバッファ
エリア(7)の確保
S12 DMAコントローラ(4-2)
に制御情報設定
S13 DMA起動
S14 受信一時バッファからメッセージ
バッファエリア(7)に転送
S15 割込み
S16 受信先の特定
S17 受信オブジェクト
起動
S18 メッセージの
読出し
送信オブジェクト (10)
カーネル (11-1)
プロセッサ モジュール (1-1)
DMAコントローラ(4-1)
DMAコントローラ(4-2)
カーネル(11-2)
プロセッサ モジュール (1-2)
受信オブジェクト(12)

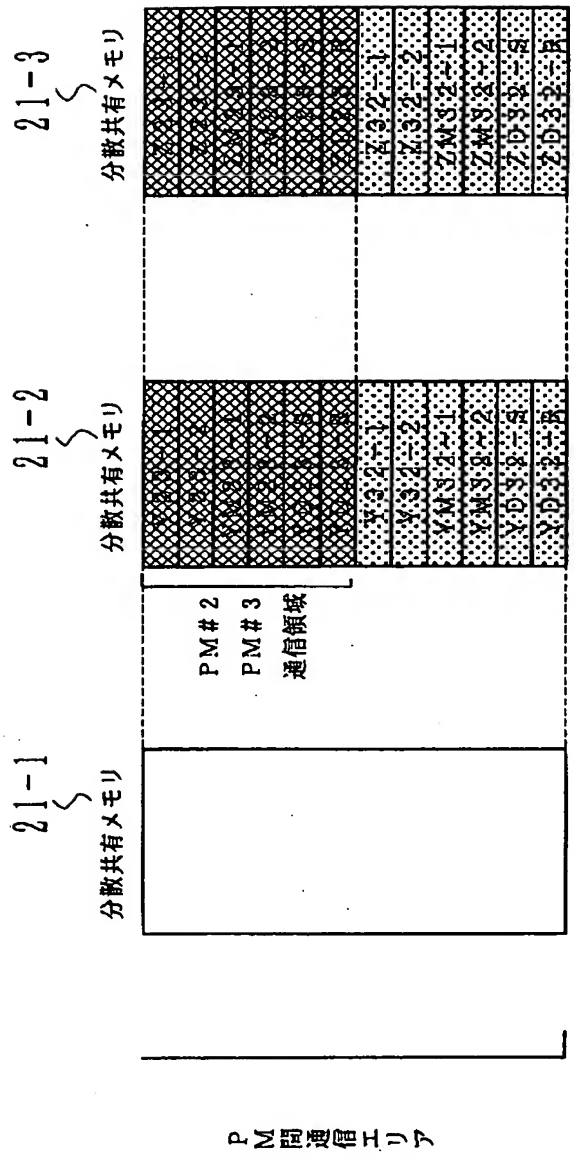
【図１８】


送信オブジェクト
プログラムウィンドウ
閉鎖期間(w1)
S100 APL
オブジェクト起動
1100
1101 MB管理マップ
XM11-2
S101 MB接続
1102
S102 メッセージ書込み
1103 メッセージ
X11-2
S103 送信処理
1104
S104 APL
オブジェクト終了
1105
受信オブジェクト
プログラムウィンドウ
開放期間(w2)
S110 APL
オブジェクト起動
1110
S111 受信処理
1111
S112 メッセージ読出し
1112 メッセージ
X11-2
S113 MB解放
1113
S114 APL
オブジェクト終了
1114 MB管理マップ
XM11-2
1115
CCR(51-1)
MCR(53-1)
送信PM (18-1)
プロセッサ (19-1)
分散共有メモリ(21-1)

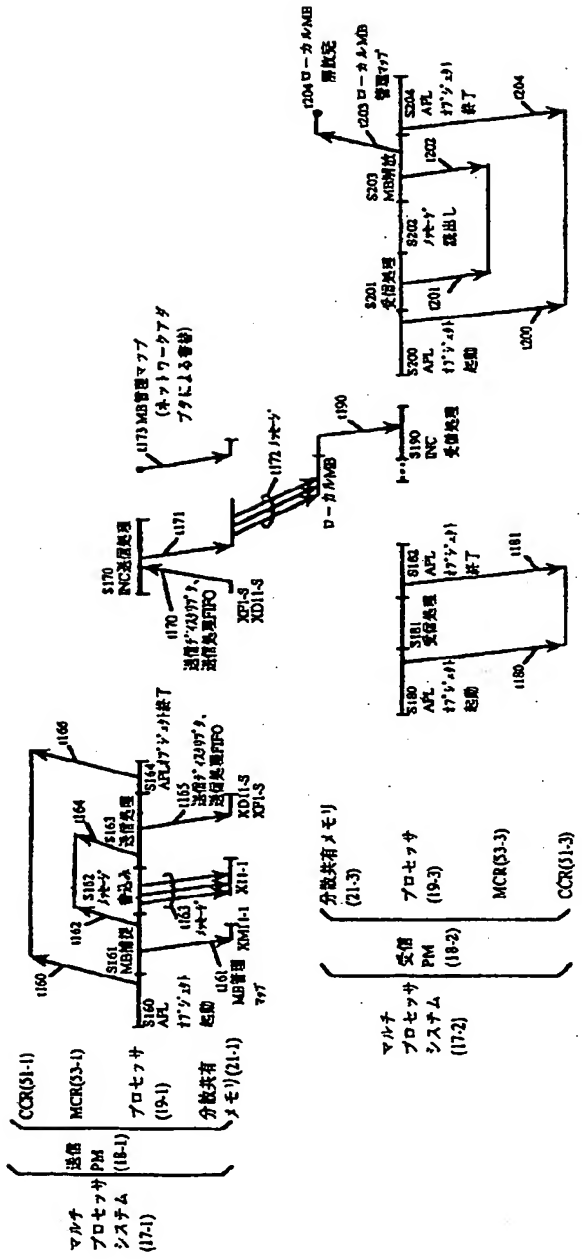
【図５】


21-1
分散共有メモリ
21-2
分散共有メモリ
21-3
分散共有メモリ
メッセージバッファ（MB）
MB管理マップ
ディスクリプタリング
X12-1
X12-2
XM12-1
XM12-2
XD12-S
XD12-R
X13-1
X13-2
XM13-1
XM13-2
XD13-S
XD13-R
X31-1
X31-2
XM31-1
XM31-2
XD31-S
XD31-R
Z13-1
Z13-2
ZM13-1
ZM13-2
ZD13-S
ZD13-R
Z31-1
Z31-2
ZM31-1
ZM31-2
ZD31-S
ZD31-R
PM#1
PM#2
通信領域
PM#2
PM#1
通信領域
PM#1
PM#3
通信領域
PM#3
PM#1
通信領域
PM間通信エリア

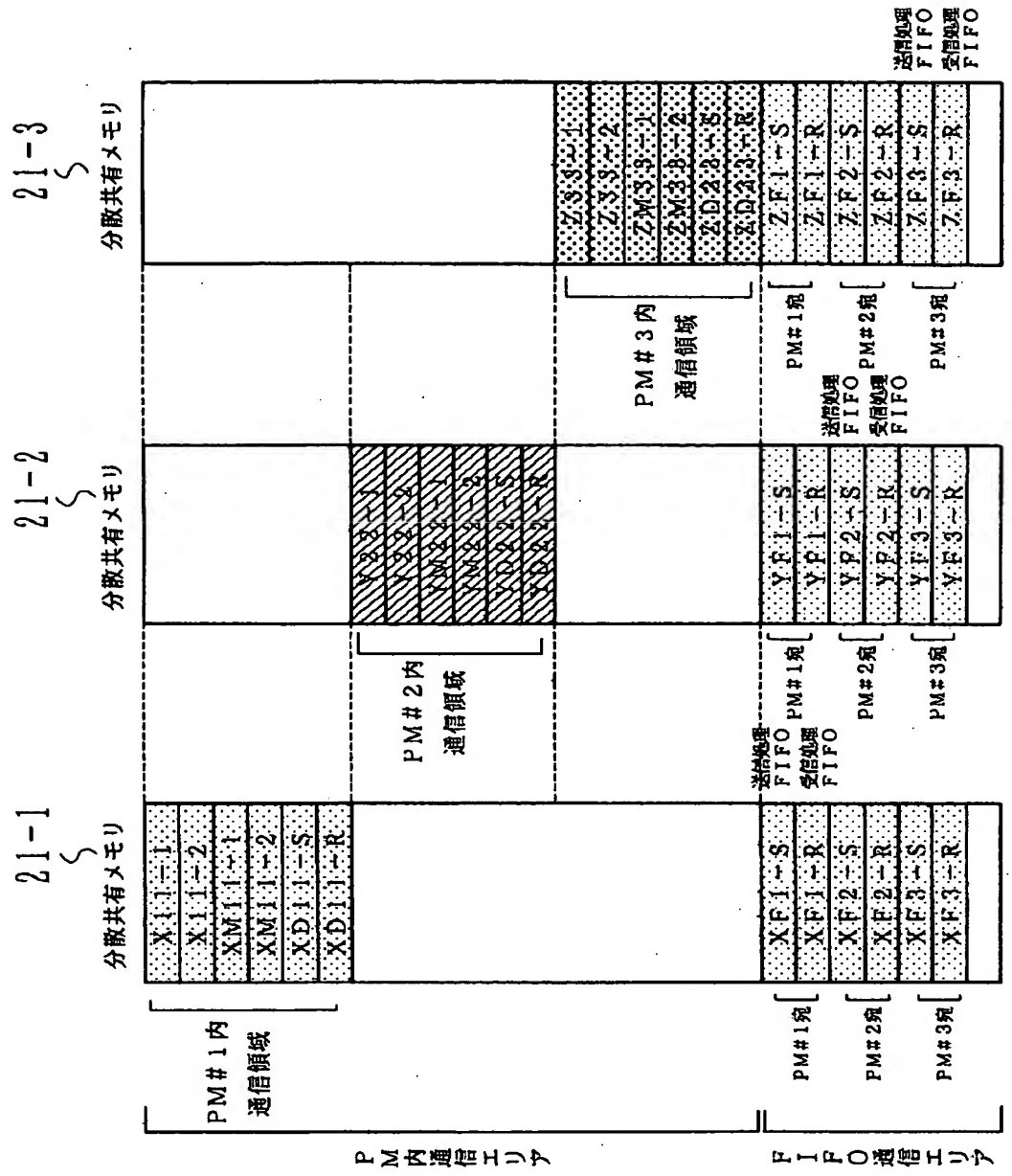
【図６】


21-1
分散共有メモリ
21-2
分散共有メモリ
21-3
分散共有メモリ
PM#2
PM#3
通信領域
Y32-1
Y32-2
YM32-1
YM32-2
YD32-S
YD32-R
PM間通信エリア

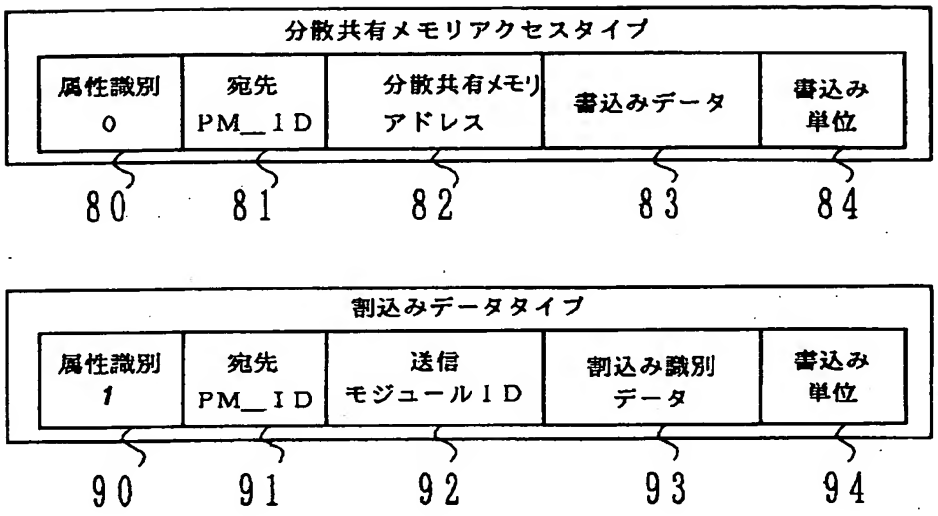
【図２０】


マルチプロセッサシステム (17-1)
CCR(51-1)
MCR(53-1)
プロセッサ (19-1)
分散共有メモリ (21-1)
送信PM (18-1)
S160 APL
S161 MB制御
S162 ハンドシェイク
S163 送信処理
S164
S165
S166
1160
1161 MB管理マップ
1162
1163 ハンドシェイク
1164
1165 送信データ, 送信処理FIFO
1166
XMI1-1
XII1-1
XDI1-S
XPI-S
マルチプロセッサシステム (17-2)
受信PM (18-2)
プロセッサ (19-3)
MCR(53-3)
CCR(51-3)
分散共有メモリ (21-3)
S180 APL
S181 受信処理
S182
1180
1181
S190 INC受信処理
S200 APL
S201 受信処理
S202 ハンドシェイク
S203 MB制御
S204
1200
1201 送出し
1202
1203 ローカルMB管理マップ
1204 ローカルMB用記憶
S170 INC送信処理
1170 送信データ, 送信処理FIFO
1171
1172 ハンドシェイク
1173 MB管理マップ (ネットワークアダプタによる参照)
ローカルMB
1190

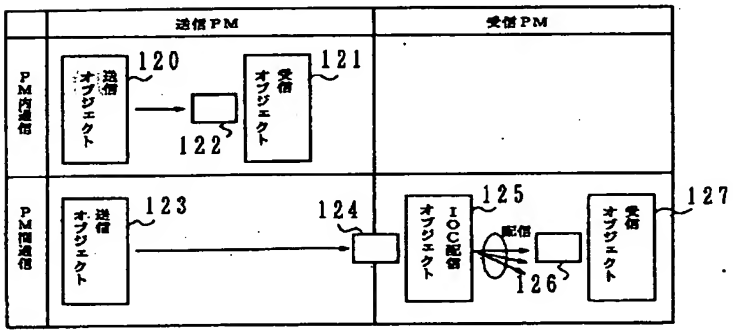
【図７】


21-1
分散共有メモリ
21-2
分散共有メモリ
21-3
分散共有メモリ
X11-1
X11-2
XM11-1
XM11-2
XD11-S
XD11-R
Y22-1
Y22-2
YM22-1
YM22-2
YD22-S
YD22-R
Z33-1
Z33-2
ZM33-1
ZM33-2
ZD23-S
ZD23-R
PM#1内
通信領域
PM#2内
通信領域
PM#3内
通信領域
XF1-S
XF1-R
XF2-S
XF2-R
XF3-S
XF3-R
YF1-S
YF1-R
YF2-S
YF2-R
YF3-S
YF3-R
ZF1-S
ZF1-R
ZF2-S
ZF2-R
ZF3-S
ZF3-R
送信処理
FIFO
受信処理
FIFO
PM#1宛
PM#2宛
PM#3宛
PM内通信エリア
FIFO通信エリア

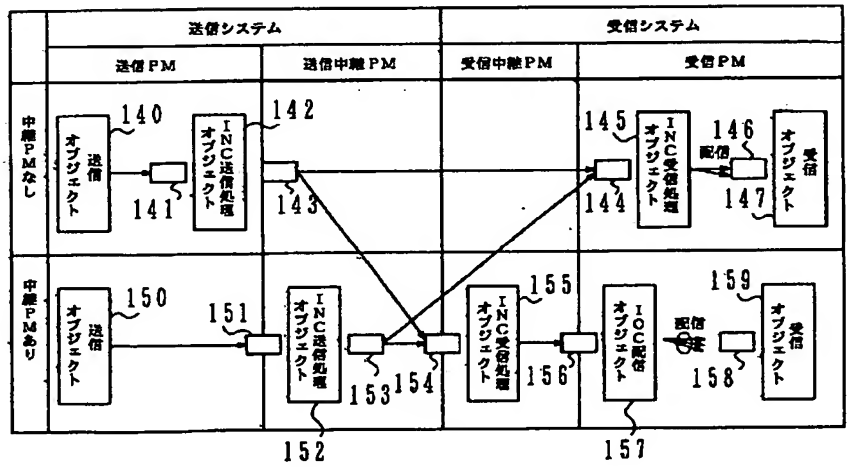
【図９】


分散共有メモリアクセスタイプ
属性識別
0
宛先
PM__ＩＤ
分散共有メモリ
アドレス
書込みデータ
書込み
単位
80
81
82
83
84
割込みデータタイプ
属性識別
1
宛先
PM__ＩＤ
送信
モジュールＩＤ
割込み識別
データ
書込み
単位
90
91
92
93
94

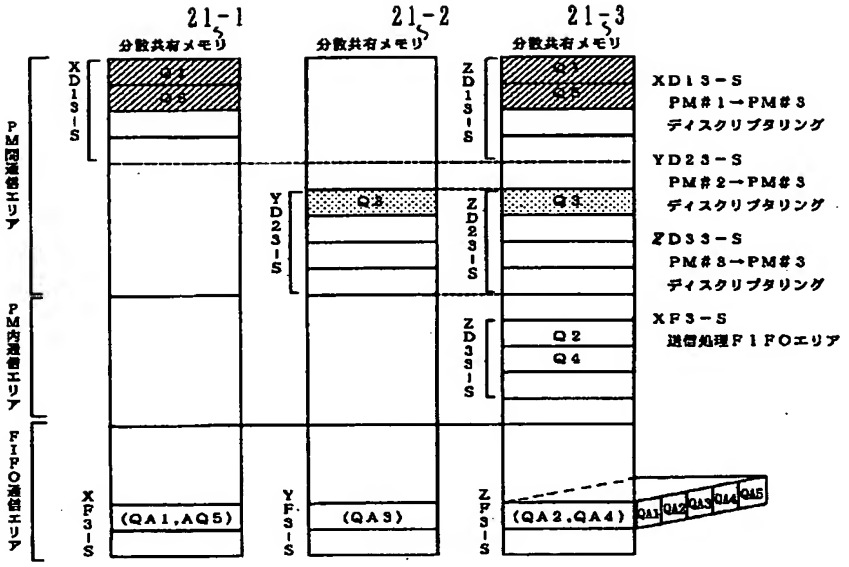
【図１０】


送信PM
受信PM
PM内通信
PM間通信
送信オブジェクト
受信オブジェクト
IOC配信オブジェクト
配信
120
121
122
123
124
125
126
127

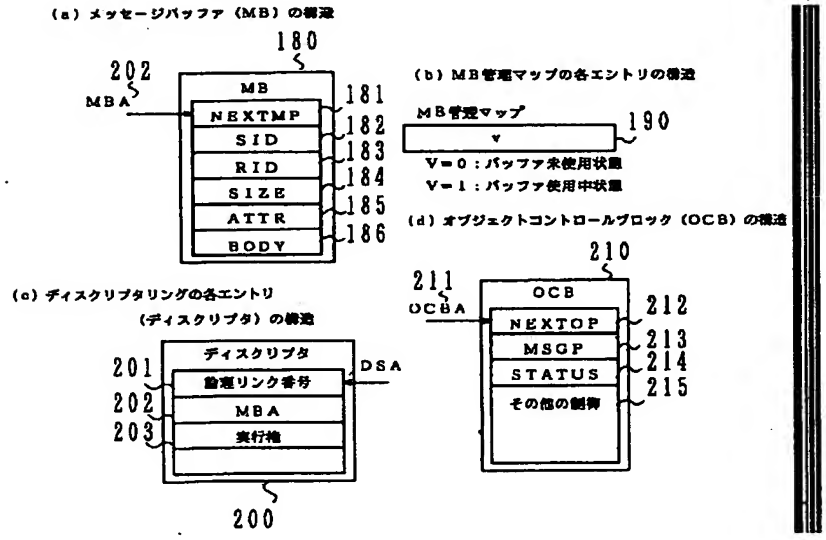
【図１１】


送信システム
受信システム
送信PM
送信中継PM
受信中継PM
受信PM
中継PMなし
中継PMあり
送信オブジェクト
INC送信処理オブジェクト
INC受信処理オブジェクト
IOC配信オブジェクト
受信オブジェクト
配信
140
141
142
143
144
145
146
147
150
151
152
153
154
155
156
157
158
159

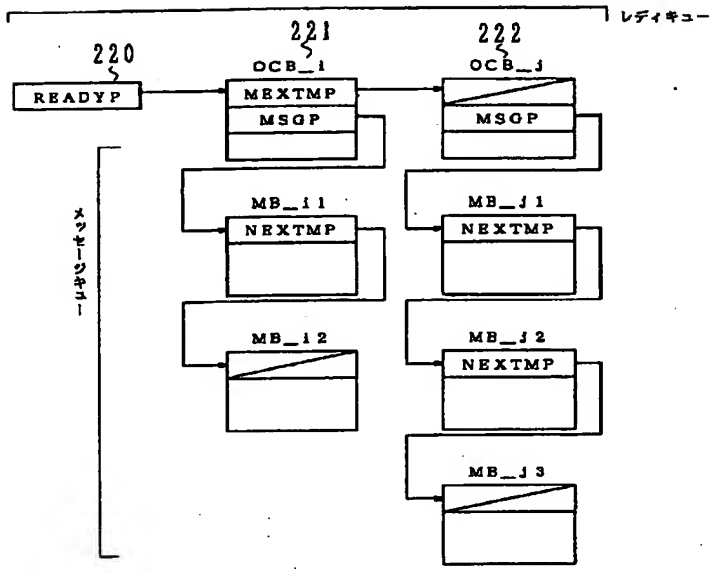
【図１３】


21-1
21-2
21-3
分散共有メモリ
分散共有メモリ
分散共有メモリ
PM間通信エリア
PM間通信エリア
FIFO通信エリア
XD13-S
YD23-S
ZD33-S
XF3-S
Q1
Q5
Q3
Q2
Q4
(QA1,AQ5)
(QA3)
(QA2,QA4)
XD13-S
PM#1→PM#3
ディスクリプタリング
YD23-S
PM#2→PM#3
ディスクリプタリング
ZD33-S
PM#3→PM#3
ディスクリプタリング
XF3-S
送信処理FIFOエリア

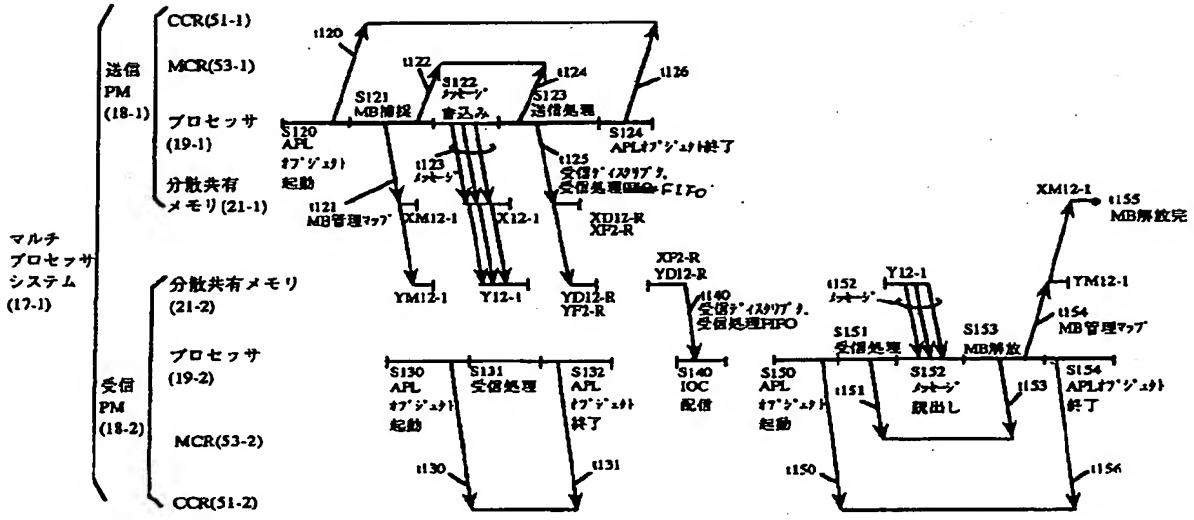
【図１５】


(a) メッセージバッファ (MB) の構造
180
202
MBA
MB
NEXTMP
SID
RID
SIZE
ATTR
BODY
181
182
183
184
185
186
(b) MB管理マップの各エントリの構造
MB管理マップ
V
190
V=0：バッファ未使用状態
V=1：バッファ使用中状態
(d) オブジェクトコントロールブロック (OCB) の構造
210
211
OCBA
OCB
NEXTOP
MSGP
STATUS
その他の制御
212
213
214
215
(c) ディスクリプタリングの各エントリ
(ディスクリプタ) の構造
ディスクリプタ
DSA
201
202
203
登録リンク番号
MBA
実行権
200

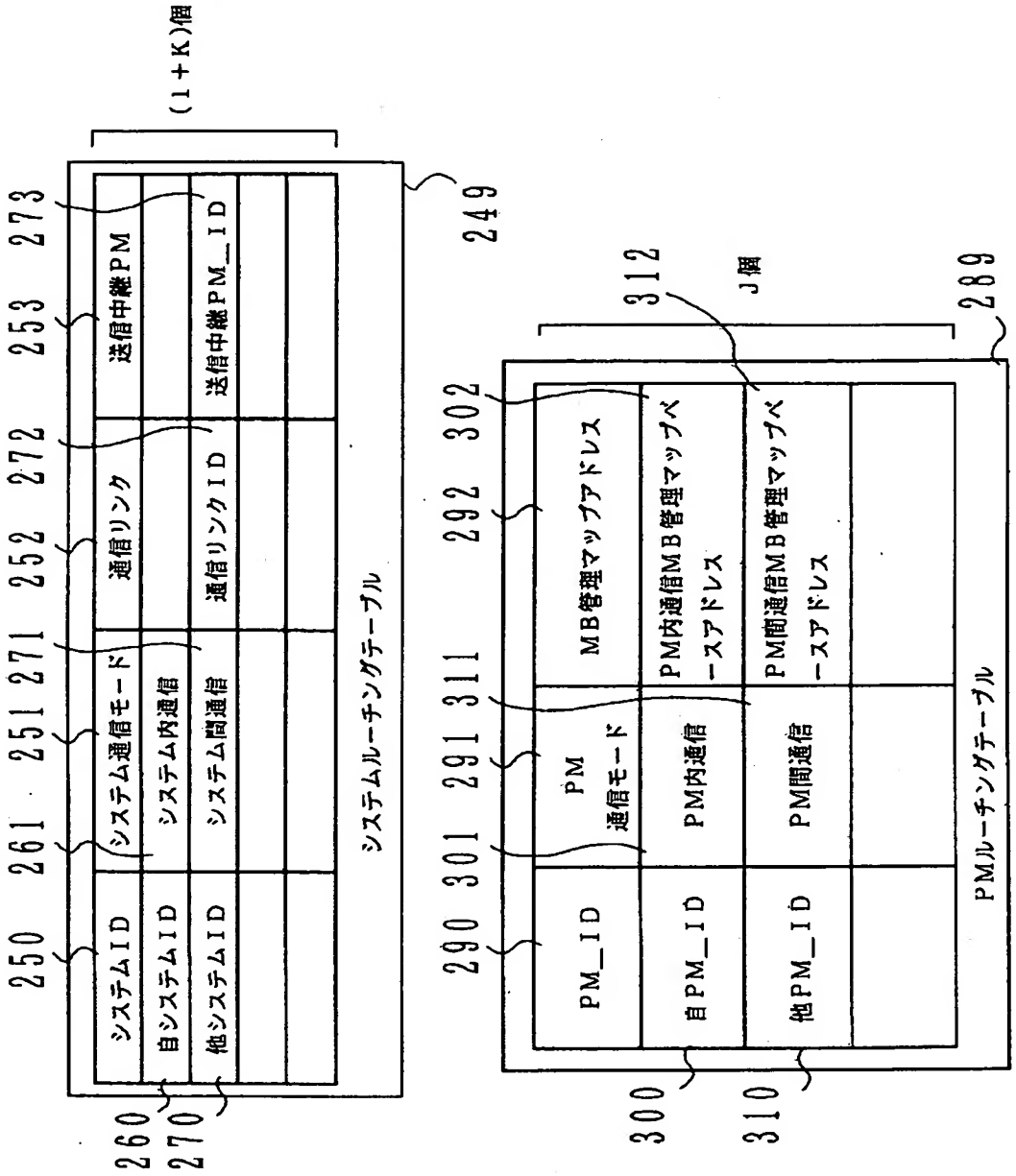
【図１６】


レディキュー
220
221
222
READYP
OCB_i
OCB_j
NEXTMP
MSGP
MB_i1
MB_i2
MB_j1
MB_j2
MB_j3
メッセージキュー

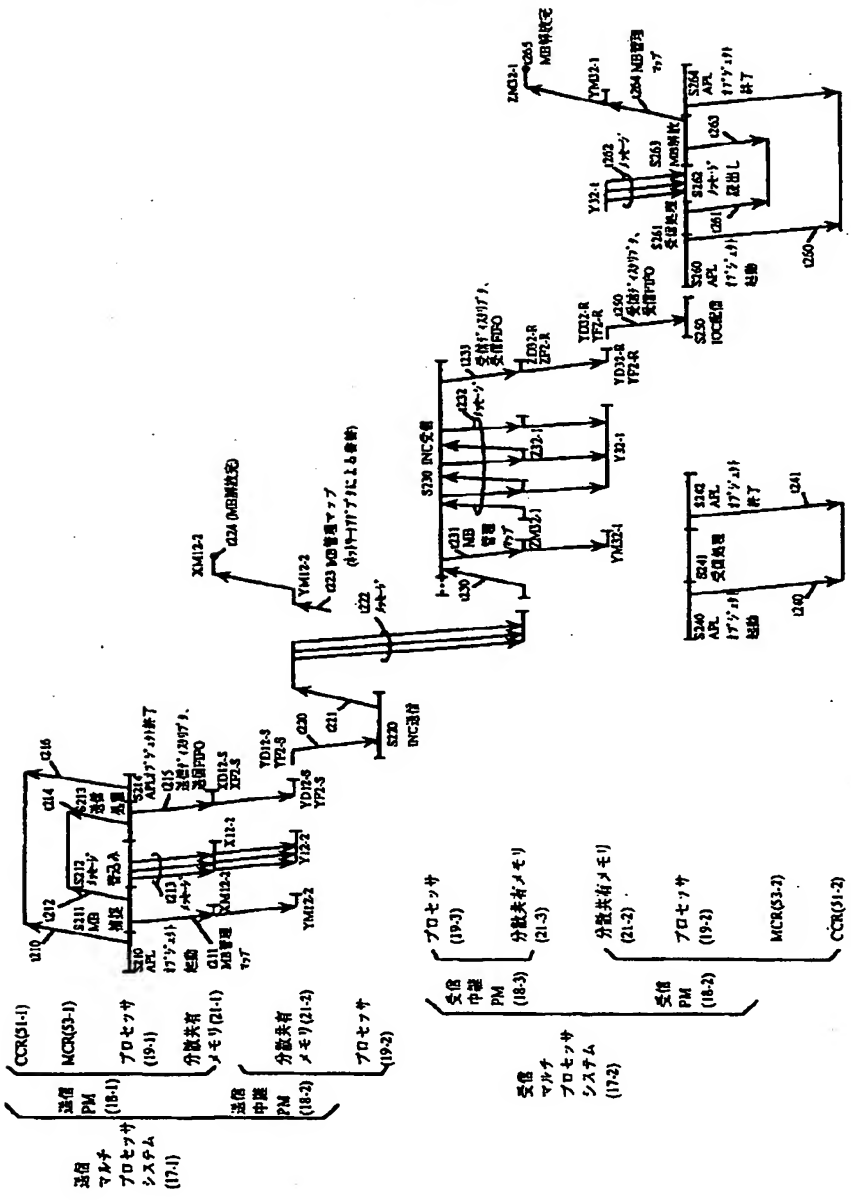
【図１９】


マルチプロセッサシステム (17-1)
送信PM (18-1)
受信PM (18-2)
CCR(51-1)
MCR(53-1)
プロセッサ (19-1)
分散共有メモリ (21-1)
分散共有メモリ (21-2)
プロセッサ (19-2)
MCR(53-2)
CCR(51-2)
S120 APL オブジェクト起動
S121 MB捕捉
S122 メッセージ書込み
S123 送信処理
S124 APLオブジェクト終了
S130 APL オブジェクト起動
S131 受信処理
S132 APL オブジェクト終了
S140 IOC 配信
S150 APL オブジェクト起動
S151 受信処理
S152 メッセージ取出し
S153 MB解放
S154 APLオブジェクト終了
t120
t121 MB管理マップ
t122
t123 メッセージ
t124
t125 受信ディスクリプタ, 受信処理FIFO
t126
t130
t131
t140 受信ディスクリプタ, 受信処理FIFO
t150
t151
t152 メッセージ
t153
t154 MB管理マップ
t155 MB解放完
t156
XM12-1
X12-1
XD12-R
XP2-R
YM12-1
Y12-1
YD12-R
YF2-R
XP2-R
YD12-R
Y12-1
YM12-1
XM12-1

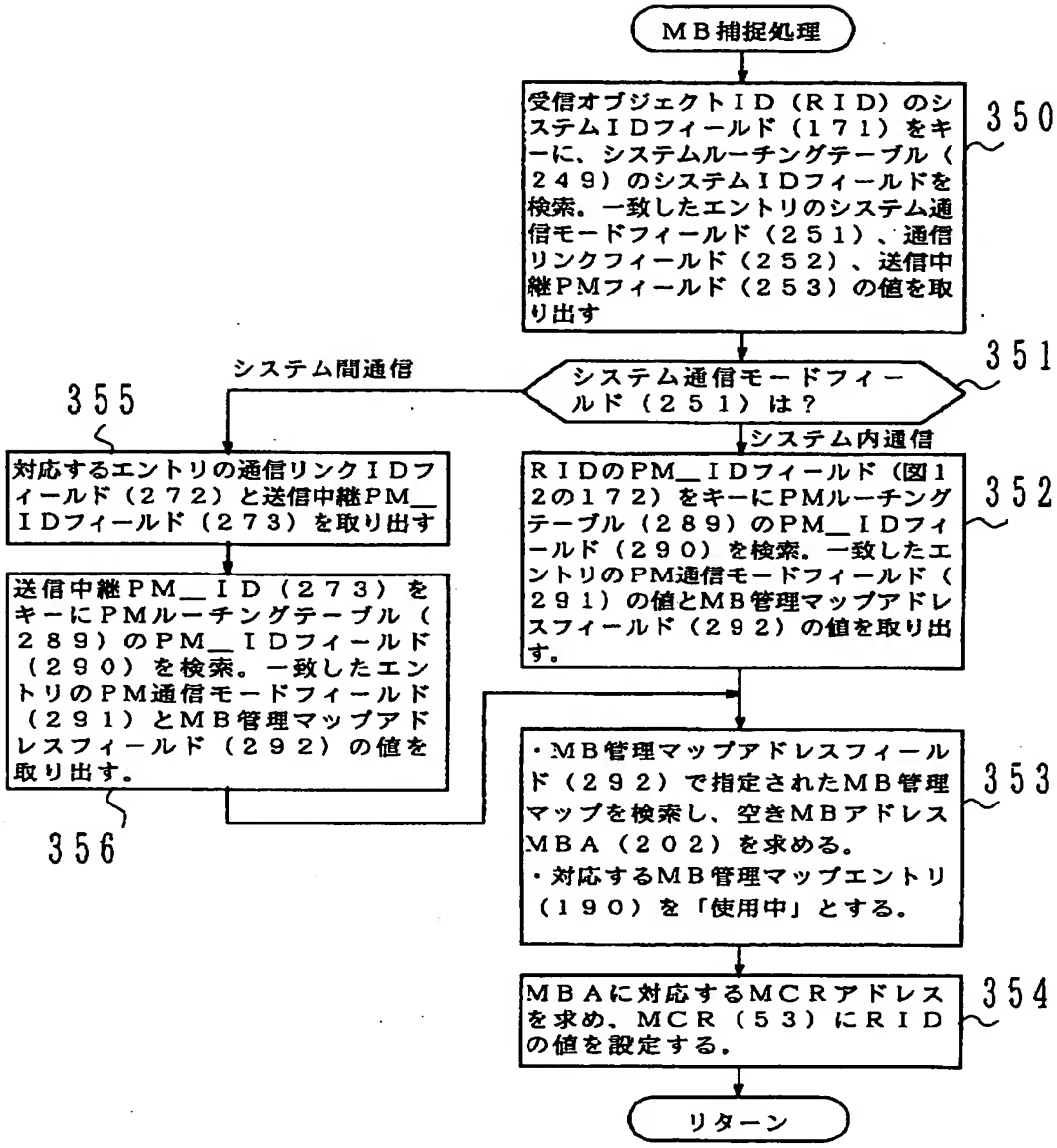
【図１７】


(1+K)個
250
261
251
271
252
272
253
273
249
システムID
自システムID
他システムID
システム通信モード
システム内通信
システム間通信
通信リンク
通信リンクID
送信中継PM
送信中継PM_ID
システムルーチングテーブル
260
270
290
301
291
311
292
302
312
J個
289
PM_ID
自PM_ID
他PM_ID
PM通信モード
PM内通信
PM間通信
MB管理マップアドレス
PM内通信MB管理マップベースアドレス
PM間通信MB管理マップベースアドレス
PMルーチングテーブル
300
310

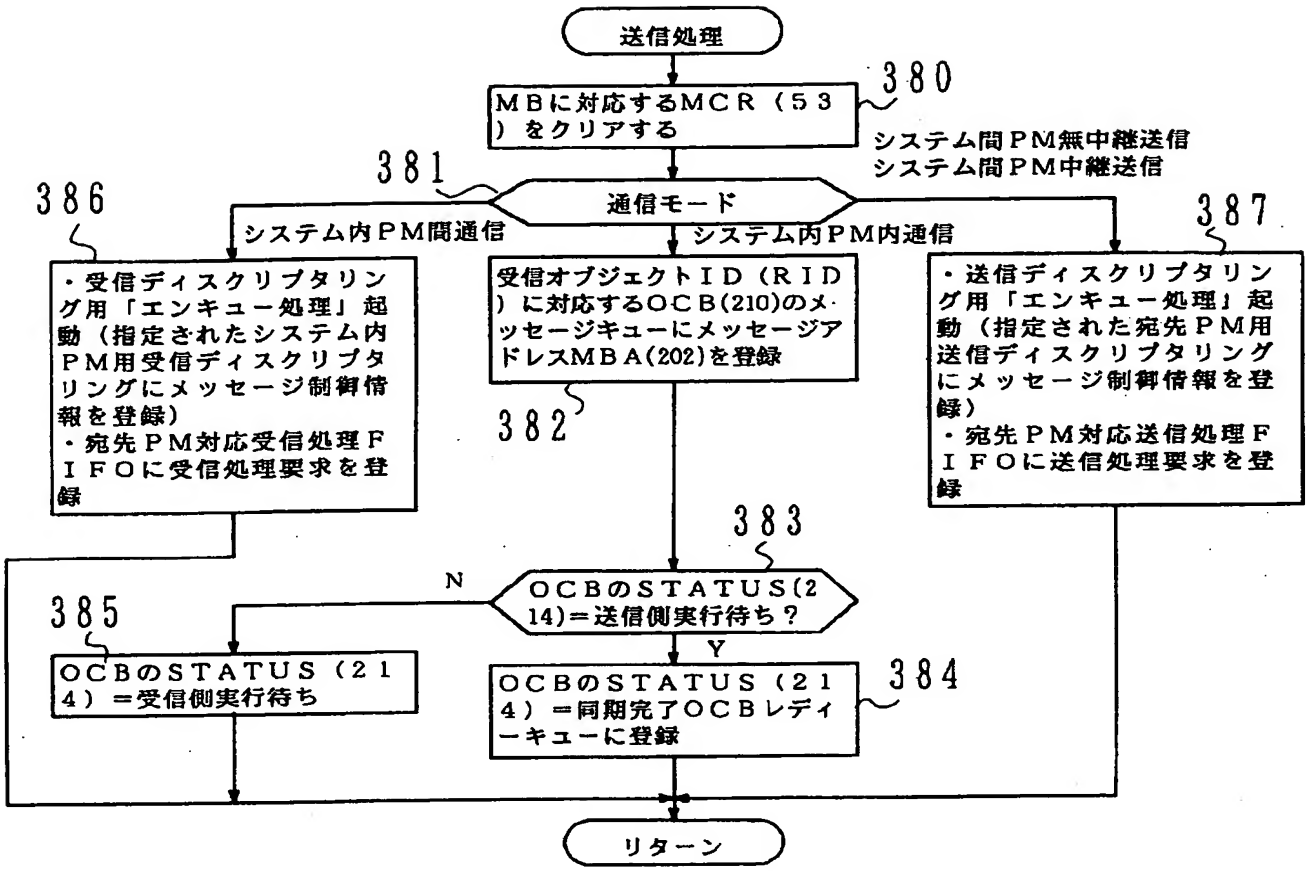
【図２１】


送信
マルチ
プロセッサ
システム
(17-1)
送信
PM
(18-1)
CCR(51-1)
MCR(53-1)
プロセッサ
(19-1)
分散共有
メモリ(21-1)
送信
中継
PM
(18-2)
分散共有
メモリ(21-2)
プロセッサ
(19-2)
受信
マルチ
プロセッサ
システム
(17-2)
受信
中継
PM
(18-3)
プロセッサ
(19-3)
分散共有メモリ
(21-3)
受信
PM
(18-2)
分散共有メモリ
(21-2)
プロセッサ
(19-2)
MCR(53-2)
CCR(51-2)

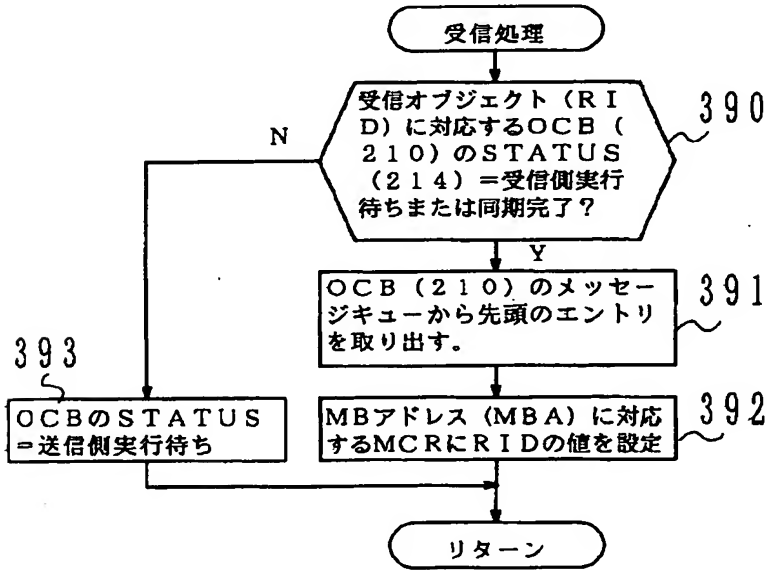
【図２３】


ＭＢ捕捉処理
受信オブジェクトＩＤ（ＲＩＤ）のシステムＩＤフィールド（１７１）をキーに、システムルーチングテーブル（２４９）のシステムＩＤフィールドを検索。一致したエントリのシステム通信モードフィールド（２５１）、通信リンクフィールド（２５２）、送信中継ＰＭフィールド（２５３）の値を取り出す
350
システム通信モードフィールド（２５１）は？
351
システム間通信
システム内通信
355
対応するエントリの通信リンクＩＤフィールド（２７２）と送信中継ＰＭ＿ＩＤフィールド（２７３）を取り出す
送信中継ＰＭ＿ＩＤ（２７３）をキーにＰＭルーチングテーブル（２８９）のＰＭ＿ＩＤフィールド（２９０）を検索。一致したエントリのＰＭ通信モードフィールド（２９１）とＭＢ管理マップアドレスフィールド（２９２）の値を取り出す。
356
ＲＩＤのＰＭ＿ＩＤフィールド（図１２の１７２）をキーにＰＭルーチングテーブル（２８９）のＰＭ＿ＩＤフィールド（２９０）を検索。一致したエントリのＰＭ通信モードフィールド（２９１）の値とＭＢ管理マップアドレスフィールド（２９２）の値を取り出す。
352
・ＭＢ管理マップアドレスフィールド（２９２）で指定されたＭＢ管理マップを検索し、空きＭＢアドレスＭＢＡ（２０２）を求める。
・対応するＭＢ管理マップエントリ（１９０）を「使用中」とする。
353
ＭＢＡに対応するＭＣＲアドレスを求め、ＭＣＲ（５３）にＲＩＤの値を設定する。
354
リターン

【図２５】

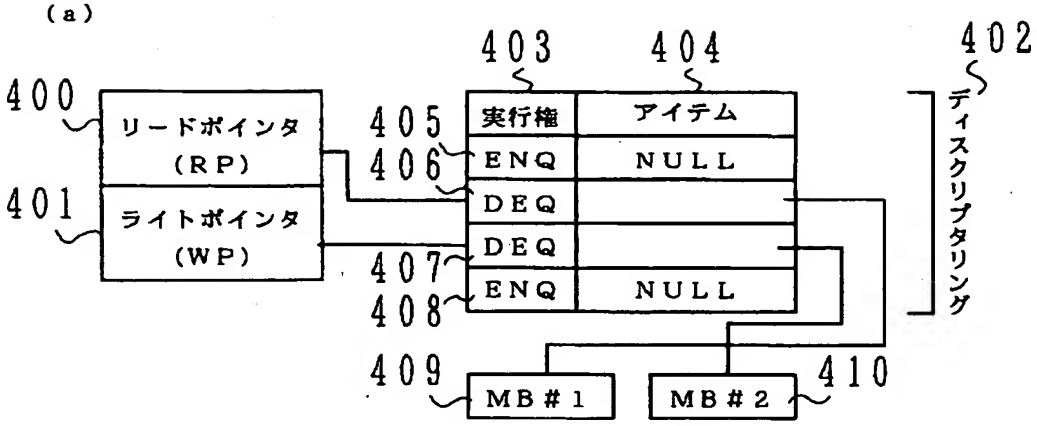

送信処理
380
ＭＢに対応するＭＣＲ（５３）をクリアする
システム間ＰＭ無中継送信
システム間ＰＭ中継送信
381
通信モード
386
システム内ＰＭ間通信
システム内ＰＭ内通信
387
・受信ディスクリプタリング用「エンキュー処理」起動（指定されたシステム内ＰＭ用受信ディスクリプタリングにメッセージ制御情報を登録）
・宛先ＰＭ対応受信処理ＦＩＦＯに受信処理要求を登録
受信オブジェクトＩＤ（ＲＩＤ）に対応するＯＣＢ(210)のメッセージキューにメッセージアドレスＭＢＡ(202)を登録
・送信ディスクリプタリング用「エンキュー処理」起動（指定された宛先ＰＭ用送信ディスクリプタリングにメッセージ制御情報を登録）
・宛先ＰＭ対応送信処理ＦＩＦＯに送信処理要求を登録
382
383
N
ＯＣＢのＳＴＡＴＵＳ(214)＝送信側実行待ち？
385
Y
ＯＣＢのＳＴＡＴＵＳ（２１４）＝受信側実行待ち
ＯＣＢのＳＴＡＴＵＳ（２１４）＝同期完了ＯＣＢレディーキューに登録
384
リターン

【図２６】

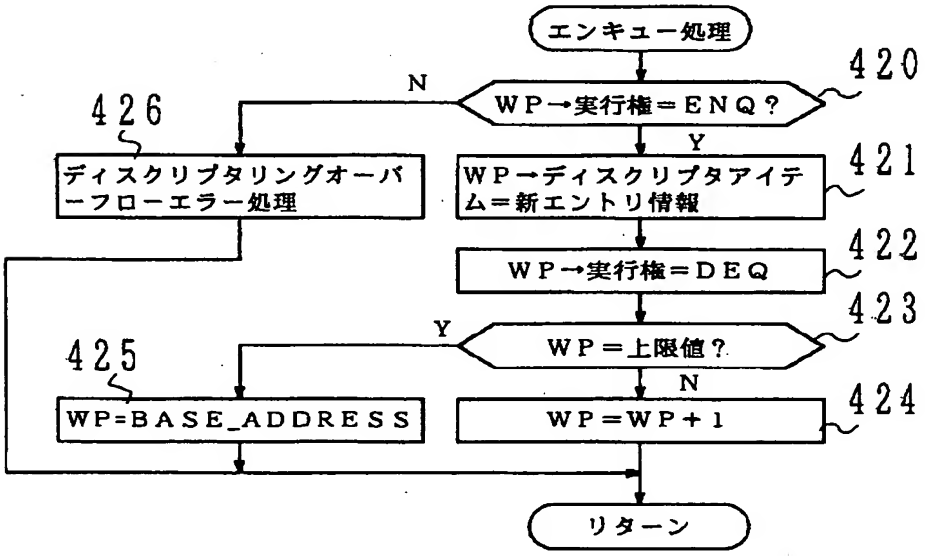

受信処理
受信オブジェクト（ＲＩＤ）に対応するＯＣＢ（２１０）のＳＴＡＴＵＳ（２１４）＝受信側実行待ちまたは同期完了？
390
N
Y
ＯＣＢ（２１０）のメッセージキューから先頭のエントリを取り出す。
391
393
ＯＣＢのＳＴＡＴＵＳ＝送信側実行待ち
ＭＢアドレス（ＭＢＡ）に対応するＭＣＲにＲＩＤの値を設定
392
リターン

【図２７】

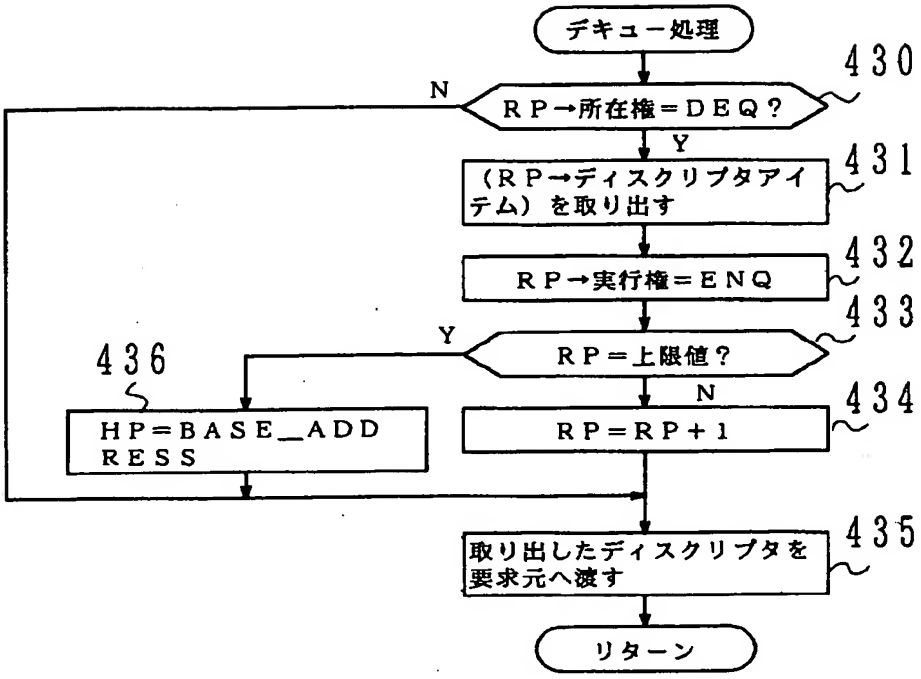
(a)


400
リードポインタ
(RP)
401
ライトポインタ
(WP)
403
404
402
ディスクリプタリング
405
406
407
408
409
410
実行権
アイテム
ENQ
NULL
DEQ
DEQ
ENQ
NULL
MB＃1
MB＃2

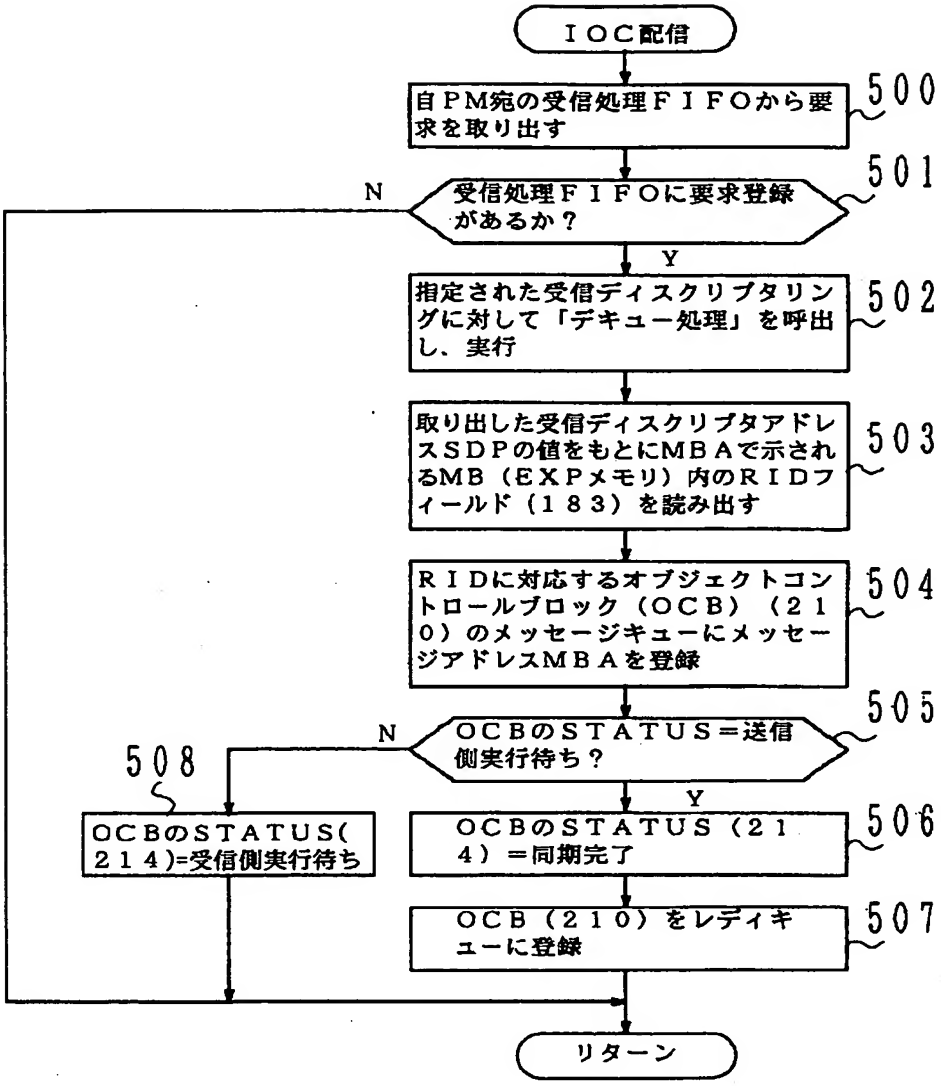
(b)


エンキュー処理
420
WP→実行権＝ENQ？
N
Y
426
ディスクリプタリングオーバーフローエラー処理
421
WP→ディスクリプタアイテム＝新エントリ情報
422
WP→実行権＝DEQ
423
WP＝上限値？
Y
N
425
WP=BASE_ADDRESS
424
WP＝WP＋1
リターン

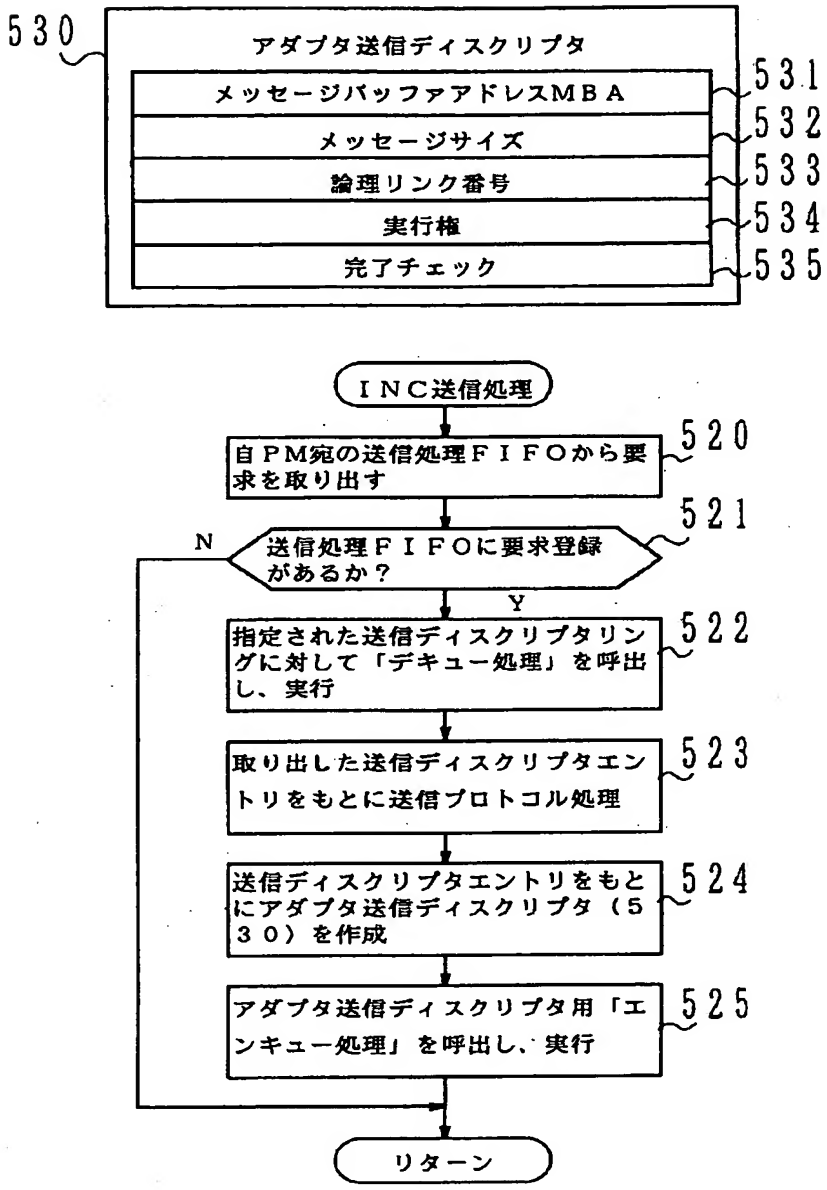
【図２８】


デキュー処理
430
N
ＲＰ→所在権＝ＤＥＱ？
Y
（ＲＰ→ディスクリプタアイテム）を取り出す
431
ＲＰ→実行権＝ＥＮＱ
432
433
Y
ＲＰ＝上限値？
436
N
434
ＨＰ＝ＢＡＳＥ＿ＡＤＤＲＥＳＳ
ＲＰ＝ＲＰ＋１
435
取り出したディスクリプタを要求元へ渡す
リターン

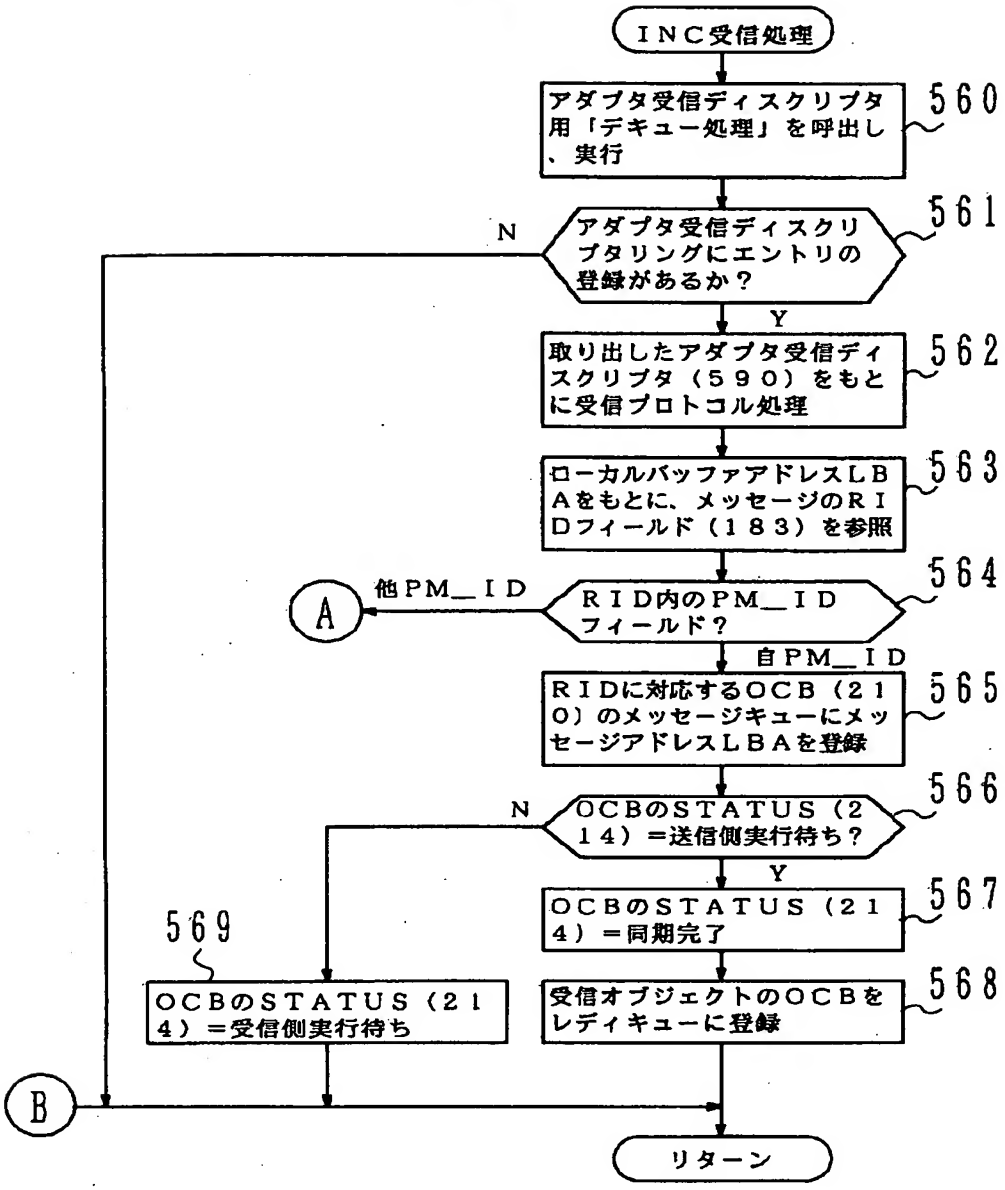
【図２９】


ＩＯＣ配信
自ＰＭ宛の受信処理ＦＩＦＯから要求を取り出す
500
受信処理ＦＩＦＯに要求登録があるか？
501
N
Y
指定された受信ディスクリプタリングに対して「デキュー処理」を呼出し、実行
502
取り出した受信ディスクリプタアドレスＳＤＰの値をもとにＭＢＡで示されるＭＢ（ＥＸＰメモリ）内のＲＩＤフィールド（１８３）を読み出す
503
ＲＩＤに対応するオブジェクトコントロールブロック（ＯＣＢ）（２１０）のメッセージキューにメッセージアドレスＭＢＡを登録
504
ＯＣＢのＳＴＡＴＵＳ＝送信側実行待ち？
505
N
Y
508
ＯＣＢのＳＴＡＴＵＳ（２１４）=受信側実行待ち
ＯＣＢのＳＴＡＴＵＳ（２１４）＝同期完了
506
ＯＣＢ（２１０）をレディキューに登録
507
リターン

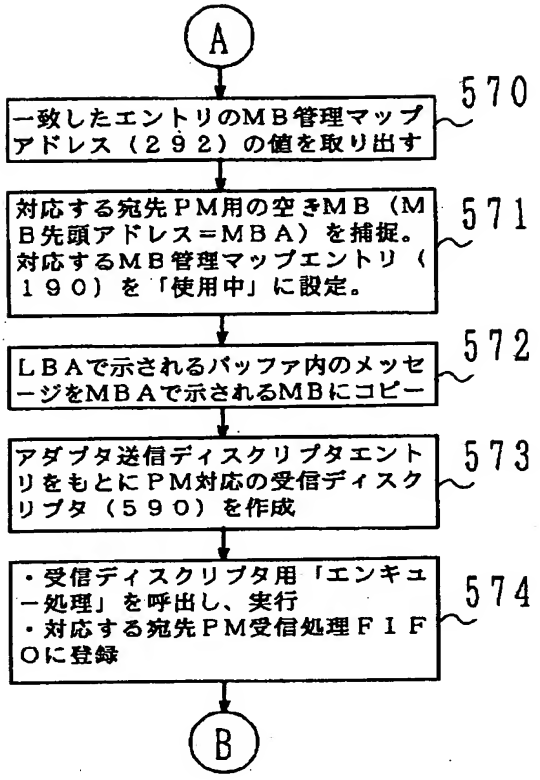
【図３０】


５３０
アダプタ送信ディスクリプタ
メッセージバッファアドレスＭＢＡ
メッセージサイズ
論理リンク番号
実行権
完了チェック
５３１
５３２
５３３
５３４
５３５
ＩＮＣ送信処理
自ＰＭ宛の送信処理ＦＩＦＯから要求を取り出す
５２０
送信処理ＦＩＦＯに要求登録があるか？
５２１
Ｎ
Ｙ
指定された送信ディスクリプタリングに対して「デキュー処理」を呼出し、実行
５２２
取り出した送信ディスクリプタエントリをもとに送信プロトコル処理
５２３
送信ディスクリプタエントリをもとにアダプタ送信ディスクリプタ（５３０）を作成
５２４
アダプタ送信ディスクリプタ用「エンキュー処理」を呼出し、実行
５２５
リターン

【図３１】


ＩＮＣ受信処理
アダプタ受信ディスクリプタ用「デキュー処理」を呼出し、実行
560
アダプタ受信ディスクリプタリングにエントリの登録があるか？
561
N
Y
取り出したアダプタ受信ディスクリプタ（５９０）をもとに受信プロトコル処理
562
ローカルバッファアドレスＬＢＡをもとに、メッセージのＲＩＤフィールド（１８３）を参照
563
ＲＩＤ内のＰＭ＿ＩＤフィールド？
564
他ＰＭ＿ＩＤ
自ＰＭ＿ＩＤ
A
ＲＩＤに対応するＯＣＢ（２１０）のメッセージキューにメッセージアドレスＬＢＡを登録
565
ＯＣＢのＳＴＡＴＵＳ（２１４）＝送信側実行待ち？
566
N
Y
ＯＣＢのＳＴＡＴＵＳ（２１４）＝同期完了
567
受信オブジェクトのＯＣＢをレディキューに登録
568
569
ＯＣＢのＳＴＡＴＵＳ（２１４）＝受信側実行待ち
B
リターン

【図３２】




---

【手続補正書】
【提出日】平成７年２月６日
【手続補正１】
【補正対象書類名】明細書
【補正対象項目名】００１５
【補正方法】変更
【補正内容】
【００１５】図５～図７のエリア内の記述に示すように、分散共有メモリ２１上の各エリアを識別するために、Ｍｉｊ－ｋのような識別名を用いている。ここで、Ｍはプロセッサモジュール対応のＭＢ／ＭＢ管理マップ／ＭＢディスクリプタリング／ＦＩＦＯ通信エリアの識別記号であり、ＰＭ（１８－１）に対応するＭＢはＸ、そのＭＢ管理マップはＸＭ、ＭＢディスクリプタリングはＸＤ、ＦＩＦＯ通信エリアはＸＦと表現される。同じように、ＰＭ（１８－２）に対応するＭＢはＹ、そのＭＢ管理マップはＹＭ、ＭＢディスクリプタリングはＹＤ、ＦＩＦＯ通信エリアはＹＦ、ＰＭ（１８－３）に対応するＭＢはＺ、そのＭＢ管理マップはＺＭ、ＭＢディスクリプタリングはＺＤ、ＦＩＦＯ通信エリアはＺＦのように名称が付けられている。また、ｉ，ｊは、それぞれ送信ＰＭの識別番号（ＰＭ　ＩＤ）と受信ＰＭの識別番号（ＰＭ　ＩＤ）を表わしており、ＰＭ＃１（１８－１）、ＰＭ＃２（１８－２）、ＰＭ＃３（１８－３）に対応してそれぞれ１，２，３の番号が付与されている。ただし、例外として、ＦＩＦＯ通信エリアＸＦ，ＹＦ，ＺＦは値ｊのみを有し、値ｉを持っていない。その理由としては、ＦＩＦＯ通信エリアが値ｊで表わされる受信側プロセッサモジュール対応に分割されており、値ｉで表わされる各送信プロセッサモジュール対応に分割されており、値ｉで表わされる各送信プロセッサモジュール間で共通に使用するため、エリアの指定情報として値ｉを必要としないためである。
【手続補正２】
【補正対象書類名】明細書
【補正対象項目名】００１８
【補正方法】変更
【補正内容】
【００１８】ディスクリプタリングエリアも、ＭＢと同じように、宛先ＰＭ－ＩＤが同一の複数のＭＢディスクリプタを分散共有メモリの連続するアドレスロケーショ

ンに割り付けており、割り付けた全体をディスクリプタリングと呼ぶ。『リング』と呼ばれている理由は、ディスクリプタリングに含まれる複数のディスクリプタを若いアドレス順にサイクリックに使用していくからである。送信側の各ディスクリプタリングと同一アドレスのエリアが、受信プロセッサモジュールの分散共有メモリにも配置される。ディスクリプタリングは、送信オブジェクトからのメッセージ制御情報を他のＰＭあるいは他のシステムに引き継ぐための『送信ディスクリプタリング』と、他のＰＭあるいは他のシストムから受信したメッセージ制御情報を受信オブジェクトに引き継ぐための『受信ディスクリプタリング』に分けられる。ディスクリプタリングもＭＢやＭＢ管理マップの場合と同じように、送信側と受信側の同一アドレス間でペアを構成する。ディスクリプタリング内のディスクリプタとＭＢとの対応は、実行時にカーネルにより動的に決定される。具体的には、例えばＰＭ＃１（１８－１）からＰＭ＃２（１８－２）にメッセージを送信する場合に、ＰＭ（１８－１）のカーネルが空きのＭＢプールの中から適当なＭＢ（例えばＸ１２－２）を捕捉し、次に対応する送信ディスクリプタリング（例えばＸ１２－Ｓ）を選択し、その中の『未使用』の最若アドレスのディスクリプタにＭＢアドレスを登録する。このような方法でディスクリプタとＭＢとの対応が動的に決定される。なお、送信ディスクリプタリングＸＤ１２－Ｓとペアを組む送信ディスクリプタリングはＹＤ１２－Ｓである。

【手続補正３】
【補正対象書類名】明細書
【補正対象項目名】００１９
【補正方法】変更
【補正内容】
【００１９】図７におけるＦＩＦＯ通信エリアは、送信オブジェクトからのメッセージ受信処理要求を他のＰＭまたは他のシステムに通知するための『送信処理ＦＩＦＯエリア』と、他のＰＭまたは他のシステムから受信したメッセージ受信処理要求を受信オブジェクトに通知するための『受信処理ＦＩＦＯエリア』に分けられる。このうち、メッセージ処理要求送信側の分散共有メモリは、通常のＲＡＭ（ランダムアクセスメモリ）で構成されるが、メッセージ処理要求受信側の分散共有メモリは、ＦＩＦＯで構成されている。例えば、ＰＭ＃１（１８－１）上のＰＭ＃１（１８－１）宛受信処理ＦＩＦＯエリアＸＦ１－Ｒは、他のＰＭからのメッセージ受信処理要求を記憶するためにＦＩＦＯメモリで構成されているが、ＰＭ＃２上のＰＭ＃１（１８－１）宛受信処理ＦＩＦＯエリアＹＦ１－Ｒおよび、ＰＭ＃３上のＰＭ＃１（１８－１）宛受信処理ＦＩＦＯエリアＺＦ１－Ｒは、いずれもＲＡＭで構成されている。この理由としては、受信処理要求を書き込む側のＰＭ（ＰＭ＃２とＰＭ＃３）が受信処理要求をそれぞれ受信処理ＦＩＦＯエリアＹＦ１－ＲとＺＦ１－Ｒに同時に書き込んだ場合、それらが同時に受信処理要求を読み出す側のＰＭ（ＰＭ＃１）の受信処理ＦＩＦＯエリアＸＦ１－Ｒに到着するので、これらの受信処理要求を全て蓄積するためにＦＩＦＯメモリを使用するからである。

---

フロントページの続き

(72)発明者　田中　聡
東京都千代田区内幸町１丁目１番６号　日本電信電話株式会社内